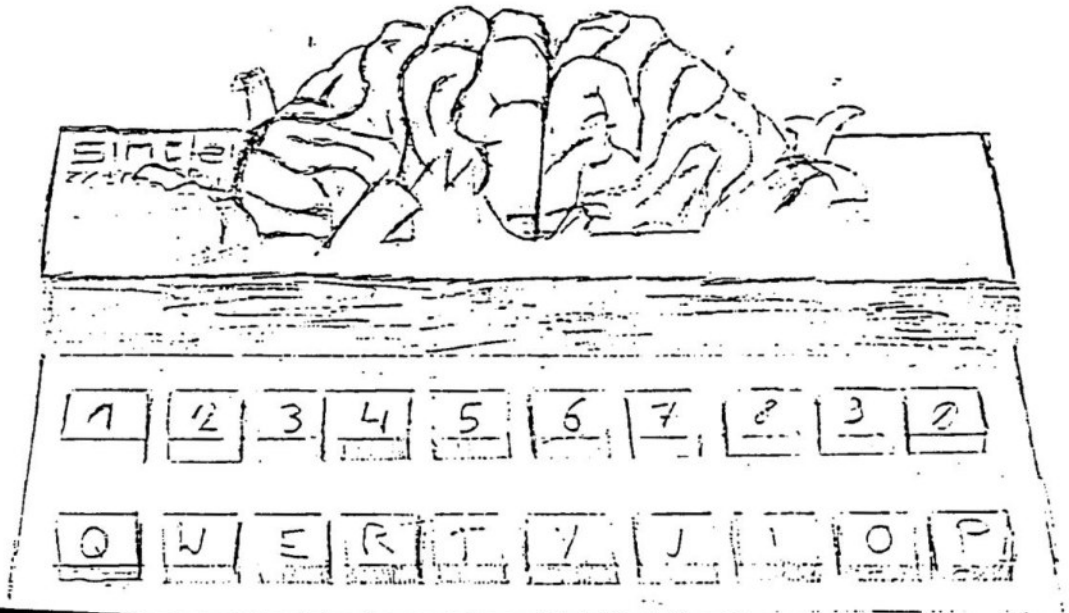
Spectrum User Club Wuppertal



Info Nr. 6/7 (78) 01. JUNI 1990

Rolf Knorre, Postfach 20 01 02, 5600 Wuppertal 2

Hallo Freunde,

der positive Trend der letzten Zeit setzt sich scheinbar fort. Auch in den letzten Wochen habe ich wieder zahlreiche Info - Beiträge erhalten. Es ist sogar so viel Material eingegangen, daß sich eines meiner Probleme von selbst erledigt hat. Damit meine ich meinen anstehenden Urlaub. Da ich vom 17.06. bis zum 07.07. nicht im Lande bin, hätte ich Ende Juni kein Info absenden können. Die Lösung liegt auf der Hand, eine Doppelnummer muß her. Genau dazu konnte ich mich durch die rege Mitarbeit entschließen. Es gibt also folgende Urlaubsregelung. Das jetzt vorliegende Juni - Info kommt einige Tage später als sonst an und hat fast doppelt soviele Seiten wie üblich. Dafür erscheint k e i n Juli - Info. Ende Juli läuft dann alles normal weiter, d.h. in den ersten August - Tagen erscheint das nächste Info mit normalem Umfang. Ich bitte alle Mitglieder um Verständnis für diese Regelung. Viel Spaß mit dem neuen Info und - an alle, die es angeht - schönen Urlaub !

Rolf Knorre

Claus-Jörg Weiske
Veit-Stoß-Str. 2
8080 Fürstenfeldbruck

25.4.90

Liebe Specci-Freunde,

heute soll es um die RAMDISK im OPUS-DISCOVERY-Betriebssystem gehen. Im Opus-DOS gibt es neben den Floppy-Disk-Laufwerken 1..4 das RAMDisk-"Laufwerk" Nr.5, das vom 48-kByte-Arbeitsspeicher des Spectrum die oberen 32 kByte abzwackt. Wer die RAMDisk 5 verwendet, hat also normalerweise nur noch 16 kByte für BASIC und Maschinenprogramme incl. Screen frei. Durch Änderung der Parameterliste für "Laufwerk" (LW) 5 läßt sich die Grenze zwischen Arbeits- und RAMDisk-Speicher verschieben.

Das OPUS-DOS 2.2/2.22 bietet zusätzlich eine RAMDisk LW6 an, die eigentlich für den Spectrum 128K gedacht ist. Hier hat man die "normalen" 48kByte als Arbeitsspeicher zur Verfügung, RAMTOP kann bei 64k bleiben. Die RAMDisk nutzt 64kByte, und weitere 16kByte werden als "Second Screen" genutzt. Durch Ändern der Parameterliste für LW6 kann man aber die RAMDisk zunächst auf Kosten des Second Screen, bei Bedarf aber auch auf Kosten des Arbeitsspeichers auf bis zu 112kByte vergrößern.

Das OPUS-DOS ist so flexibel, daß LW6 nicht nur mit dem Spectrum 128K nutzbar ist, sondern auch mit nahezu beliebigen, z.B. selbstgebauten Speichererweiterungen - wenn nur das "Bank-Switching" über I/O-Befehle (OUT adresse, banknummer) steuerbar ist. Das ist genau das Richtige für Leute, die einen umgebauten "Spectrum 80K" oder "-272K" haben! Die Speichererweiterung darf aber aus nicht mehr als 8 Bänken bestehen.

Eine RAMDisk wird wie eine Floppy benutzt. Sie muß zunächst mit FORMAT lw,"name" eingerichtet und kann dann mit CAT, SAVE*, LOAD*, OPEN#, PRINT#, INPUT#, CLOSE# usw. bedient werden. Sie ist wesentlich schneller als ein Floppy-LW. So braucht das "Vollschreiben" einer 224-kByte-RAMDisk mit ca. 30 Dateien etwa 1/3 der Zeit, die eine Floppy benötigt. Bei sehr vielen kleinen Dateien ist der Geschwindigkeitsvorteil größer, bei wenigen großen Dateien kleiner. Aber natürlich ist auch die schnellste RAMDisk nicht annähernd so schnell wie ein reines Maschinenprogramm mit LDIR-Anweisung. Letzteres ist aber nicht immer praktisch und wird komplizierter, wenn Daten zwischen unterschiedlichen Bänken transferiert werden sollen. Im Gegensatz zur Floppy geht der RAMDisk-Inhalt beim Ausschalten des Rechners natürlich verloren, es sei denn, man hat ein batteriegepuffertes statisches RAM. Trotzdem ist eine RAMDisk äußerst praktisch bei Programmen, die viel Dateiarbeit leisten oder dauernd nachladen (z.B. Tasword3 oder Nossy). Am Schluß der Arbeitssitzung kann man ja alle wichtigen Dateien mit einem Hilfsprogramm auf Floppy "rüberschaufeln". Der Befehl MOVE "d";6 TO "d";1 (o.ä.) ist hier leider nicht anwendbar, weil man ganze "Disketten" nur dann kopieren kann, wenn sie gleiches Format haben.

Wenn man die Parameterlisten im OPUS-DOS ändern will, muß das 2-kByte-RAM (6116) in der OPUS bestückt sein! Das Praktische am OPUS-DOS ist, daß die Parameterlisten zunächst im EPROM schlummern, beim ersten DOS-

Zugriff aber sofort ins RAM (falls vorhanden) kopiert werden. Dort kann man sie vom BASIC aus erreichen und ändern. Wenn man sich endlich für eine endgültige Liste entschieden hat, kann man natürlich das EPROM entsprechend brennen und spart sich so das sonst jedesmal nach dem Einschalten nötige Ändern der Liste. Aber Achtung bei geänderten EPROMs: das OPUS-DOS macht einen Quersummen-Test über das gesamte EPROM! Die Routine beginnt auf Adresse $1480 (2.1) bzw. $1554 (2.2). Wen man auch nur 1 Byte im EPROM ändert, muß man entweder den Quersummen-Eintrag (2Bytes, Adr. $1481 bzw. $1555) korrigieren oder aber das gesamte Quersummen-Programm außer Betrieb setzen. Ich habe damit keine Erfahrung, da ich am EPROM noch nicht manipuliert habe. Vielleicht meldet sich ein anderer OPUS-User zu diesem Punkt einmal im Info!

Wie man an die Laufwerk-Listen mit BASIC herankommt, hat Axel Töpfer für die Floppy-Laufwerke 1 und 2 im Info 3/88 hervorragend beschrieben (Schade, daß man von ihm im Info nichts mehr hört; ich habe sehr viel von ihm gelernt). Für LW 5 und 6 gelten lediglich andere Tabellen -Offsets. Ich gebe daher hier nur noch die BASIC-Befehle wieder, ohne sie ausführlich zu erläutern. Zur Erinnerung: das Sinclair-ROM/RAM wird mit PEEK und POKE, das OPUS-ROM/RAM mit POINT# und Datei-Befehlen angesprochen.

Die RAMDISK "Laufwerk" 5:

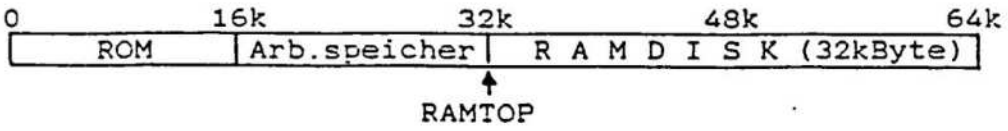
Folgendes Programm macht die Parameterliste für LW5 zugänglich:

```
  1 GO TO 500
 50 DEF FN a()=CODE INKEY$ #4+256*CODE INKEY$ #4: REM 2-Byte-Adresse im RAM lesen

100 DEF FN h$(x)=CHR$ (x+48+7*(x>9)):                REM ¦
110 LET hh=INT (d/16): LET hl=d-16*INT (d/16): REM ¦ DEZ=>HEX-
120 LET h$=FN h$(hh)+ FN h$(hl):                     REM ¦ Wandlung
140 RETURN:                                          REM ¦

500 OPEN #4;"CODE ":       REM OPUS-Speicher zugänglich machen
510 POINT #4;8192:         REM zeige auf RAM-Anfang: hier steht die Adresse der "main table"
520 POINT #4; FN a()+10:   REM zeige auf 10. Byte der Main table=Adr. der "disk info table 0A"
530 POINT #4; FN a()+13:   REM zeige auf 13. Byte d. Disk Info T.=Adr der "RAMDisk subtable"(LW5)
540 POINT #4; FN a():      REM zeige auf Anfang der "RAMDISK Subtable", also unsere Par.liste

600 FOR i=1 TO 7:          REM sie besteht aus 7 Bytes
610 LET d=CODE INKEY$ #4: GO SUB 110
620 PRINT h$;" ";          REM hexadezimal ausdrucken
630 NEXT i
700 CLOSE #4
```

Im Original-DOS sieht die Tabelle so aus:

Byte-Nr.	Inhalt (hex)	Bedeutung:	
0/1	00 80	DEFW 8000	=32768 = Anfangsadresse der RAMDisk
2/3	20 00	DEFW 0020	=Blockgröße 32 Bytes
4/5	00 04	DEFW 0400	=Anzahl der Blöcke = 1024
6	07	DEFB 07	=7+1 = 8 Blöcke f. Katalog reservieren

Das entspricht folgender Speichereinteilung im Specci:



Wir können nun z.B. die Anfangsadresse der RAMDisk ändern, wobei die Block-Anzahl bzw. -Größe entsprechend angepaßt werden muß. RAMTOP muß natürlich vorher vom BASIC aus mit CLEAR ..... angepaßt werden. Das OPUS-DOS prüft, ob die Anfangsadressse der RAMDisk größer als RAMTOP ist. Falls nicht, gibt's eine Fehlermeldung ("Disk I/O-Error").

Hier ein Programm zur Änderung der Parameterliste (Beispiel: Startadresse 48952 = $C000; Größe: 512 = $0200 Blöcke):

```
 50 DEF FN a()=CODE INKEY$ #4+256*CODE INKEY$ #4

500 OPEN #4;"CODE "
510 POINT #4;8192
520 POINT #4; FN a()+10
530 POINT #4; FN a()+13
540 POINT #4; FN a()

590 RESTORE
600 FOR i=1 TO 7:       REM 7 Bytes in der DATA-Liste
610 READ x
620 PRINT #4; CHR$ x:  REM hier wird die neue Liste "gePOKEt"
650 NEXT i
700 CLOSE #4

000 DATA  0, 192, 32,  0,  0,  2,  7 : REM dez
011 REM  00,  C0, 20, 00, 00, 02, 07       hex
012 REM   49152 ,   32  ,  512  ,  7       dez, aufgeschlüsselt
```

Leider müssen die Parameter-Bytes dezimal eingegeben werden, man kann es sicher eleganter programmieren.

Die RAMDISK "Laufwerk" 6: (nur DOS 2.2 und höher)

LW6 ist, wie erwähnt, ursprüglich für den Spectrum 128K gedacht. Aus der Parameterliste können wir also Einsicht in die Speichereinteilung dieses Specci-Typs gewinnen. Folgendes Programm bringt uns in die Parameter-Liste ("RAMDisk Subtable of Table 0A"):

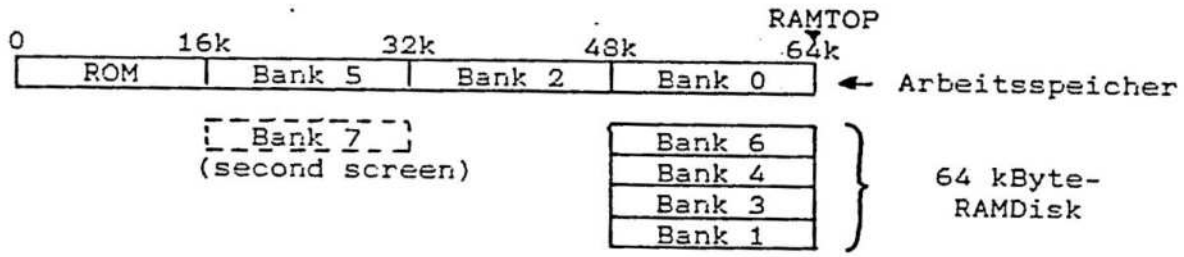
```
  1 GO TO 500
 50 DEF FN a()=CODE INKEY$ #4+256*CODE INKEY$ #4: REM 2-Byte-Adresse im RAM lesen

100 DEF FN h$(x)=CHR$ (x+48+7*(x>9)):              REM ¦
110 LET hh=INT (d/16): LET hl=d-16*INT (d/16): REM ¦ DEZ=>HEX-
120 LET h$=FN h$(hh)+ FN h$(hl):                   REM ¦ Wandlung
140 RETURN:                                        REM ¦

500 OPEN #4;"CODE ":      REM OPUS-Speicher zugänglich machen
510 POINT #4;8192:        REM zeige auf RAM-Anfang: hier steht die Adresse der "main table"
520 POINT #4; FN a()+10:  REM zeige auf 10. Byte der Main table=Adr. der "disk info table 0A"
530 POINT #4; FN a()+16:  REM zeige auf 16. Byte d. Disk Info T.=Adr. der "RAMDisk subtable"(LW6)
540 POINT #4; FN a():     REM zeige auf Anfang der "RAMDISK Subtable", also unsere Par.liste
```

```
600 FOR i=1 TO 18:          REM sie besteht aus 18 Bytes
610 LET d=CODE INKEY$ #4: GO SUB 110
620 PRINT h$;" ";           REM hexadezimal ausdrucken
630 NEXT i
700 CLOSE #4
```

Damit erhalten wir folgende Tabelle:

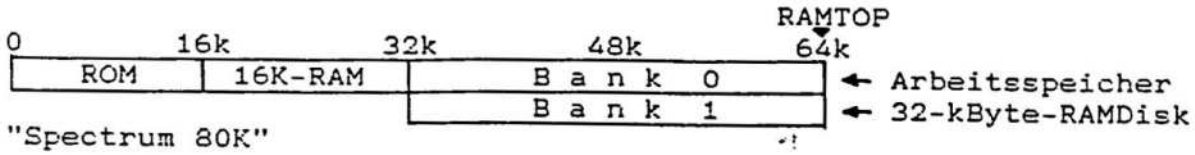
Byte-Nr.	Inhalt (hex)	Bedeutung:	
0/1	00 C0	DEFW C000	=49152 = Anfangsadresse der RAMDisk
2/3	00 01	DEFW 0100	=Blockgröße 256 Bytes
4/5	00 01	DEFW 0100	=Anzahl der Blöcke = 256
6	03	DEFB 03	=3+1 = 4 Blöcke für CATalog reservieren
7	02	DEFB 02	=Organisationsparameter (s. Text)
8/9	FD 7F	DEFW 7FFD	=Portadresse f. Bankumschaltung wichtig ist meist nur Byte8 = 253
10	10	DEFB 10	=Speicherbank, die im BASIC aktiv ist ("current page"): Bank 0
11	16	DEFB 16	=erste RAMDisk-Bank = Bank 6
12	14	DEFB 14	=2. RAMDisk-Bank = Bank 4
13	13	DEFB 13	=3. RAMDisk-Bank = Bank 3
14	11	DEFB 11	=4. RAMDisk-Bank = Bank 1
15	17	DEFB 17	=evtl. 5. RAMDisk-Bank = Bank 7
16	10	DEFB 10	=evtl. 6. RAMDisk-Bank = Bank 0
17	12	DEFB 12	=evtl. 7. RAMDisk-Bank = Bank 2

Die meisten Tabelleneinträge erklären sich selbst. Byte 7 ist ein Beweis für die flexible Konzeption des OPUS-DOS und wird später anhand eigener Speichererweiterungen beschrieben. Die Bytes 10 bis 17 geben im rechten Nibble (Halbbyte) die Bank-Nr. an. Bank 0 gehört normalerweise zum BASIC-Arbeitsspeicher. Das linke Nibble ist immer=1 und steuert (auf mir unbekannte Weise) den 48k/128k-Modus des Spectrum 128K und ist bei Selbstbau-Speichererweiterungen uninteressant. Hier die aus dieser Tabelle (und dem ROM-Listing von Marcel van Dongen) resultierende Speichereinteilung des Spectrum 128K:

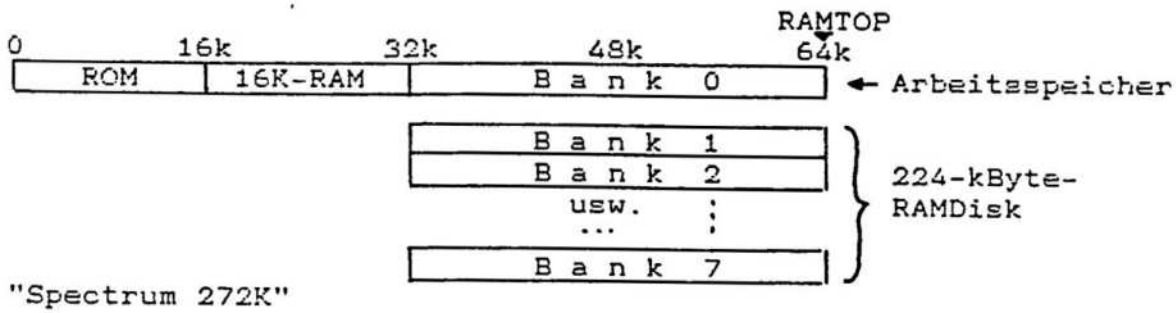


Deutlich wird die Aufteilung des Speichers in 16-kByte-Blöcke. Die Bänke 7, 0 und 2 können der RAMDisk zugeschlagen werden, wenn man zunächst auf den "second screen" verzichtet und weiters RAMTOP herabsetzt. Die Bytes 2..5 in der Tabelle sind dann zu ändern, um die RAMDisk entsprechend zu vergrößern. Wie man sieht, fehlt Bank 5 in der Liste. Bank 5 kann natürlich nicht für die RAMDisk verwendet werden, weil sie für das Specci-Betriebssystem (Screen, Systemvariable, minimaler Arbeitsspeicher) benötigt wird! In LW6 wird übrigens nicht, wie in LW5, RAMTOP überprüft - also aufpassen! Die Tabellenübersicht gibt Vorschläge für die Änderungen in der Parametertabelle.

Wie können wir uns nun die Parametertabelle für eigene Speichererweiterungen zunutze machen? Dazu sollten wir zunächst mal unsere Speichereinteilung aufzeichnen. Hier als Beispiele die Einteilung für den "Spectrum 80K" und den "Spectrum 272K" (im Mai-Info beschrieben):



"Spectrum 80K"



"Spectrum 272K"

In beiden Fällen beginnt der Bank-Speicher bei 32768. RAMTOP bleibt unverändert bei 64k, d.h. der BASIC-Arbeitsspeicher wird durch die RAMDisk nicht verkleinert. Beim Specci 80K kann aber die RAMDisk auf 64 kByte vergrößert werden, wenn man RAMTOP auf 32k oder kleiner setzt. Wichtig ist, daß der Bank-Speicher (wegen der max. Tabellenlänge) incl. Bank 0 aus nicht mehr als 8 Bänken besteht. Hier eine Tabellenübersicht für verschiedene Specci-Varianten und RAMDisk-Größen:

	Specci-Type	128 K				272K	80 K	
Byte -Nr. ↓	RAMDisk-Größe RAMTOP (max.) BASIC-Sp. Sec. Screen	64 k 64 k 48 k ja	80 k 64 k 48 k nein	96 k 48 k 32 k nein	112k 32 k 16 k nein	222k 64 k 48 k -	32 k 64 k 48 k -	64 k 32 k 32 k -
1/0	Anfangsadr.	C000	C000	C000	C000	8000	8000	8000
3/2	Blockgröße	0100	0100	0100	0100	0100	0100	0100
5/4	Anz. Blöcke	0100	0140	0180	01C0	0380	0080	0100
6	Anz.CAT-Bl.-1	03	03	04	04	06	02	03
7	Org.Parameter	02	02	02	02	01	01	01
9/8	Port-Adresse	7FFD	7FFD	7FFD	7FFD	7FFD	7FFD	7FFD
10	Bank i.Arb.sp.	10	10	(10)	(10)	00	00	(00)
11	1. RAMD.Bank	16	16	16	16	01	01	01
12	2. "	14	14	14	14	02	(00)	00
13	3. "	13	13	13	13	03	(--)	(--)
14	4. "	11	11	11	11	04	(--)	(--)
15	5. "	(17)	17	17	17	05	(--)	(--)
16	6. "	(10)	(10)	10	10	06	(--)	(--)
17	7. "	(12)	(12)	(12)	12	07	(--)	(--)

In Klammern gesetzte Bytes werden bei der jeweiligen RAMDisk-Größe nicht verwendet. Über eine günstige Vorgabe für Blockgröße und Anzahl der CATalog-Blöcke kann man bei Axel Töpfer (Info 3/88) und in Beschreibungen von "Disk-Doktor"-Programmen nachlesen. Zur Bank-Umschaltung liest das OPUS-DOS aus der Parametertabelle die Portadresse (Bytes 8 u. 9), gibt Byte 8 ins C-Register und die Bank-Nr. in

den Akkumulator und schaltet dann die benötigte Bank mit "OUT (C),A" ein (IORQ=low). Auf diesen Befehl muß die selbstgebaute Speichererweiterung ansprechen.

Bis jetzt bin ich noch die Erklärung für den "Organisationsparameter", Byte 7, schuldig geblieben. Diese Byte macht den Einsatz von LW6 für Selbstbau-Speichererweiterungen überhaupt erst möglich, weil es die hardwaremäßige Speicherorganisation beschreibt. In dem sehr gut kommentierten OPUS-ROM-Listing von M. van Dongen ist es etwas ungenau als "multiplication byte" beschrieben, aber sein Aufbau geht aus der RAMDisk-Routine im DOS hervor. Diese Routine möchte ich hier nicht im einzelnen erläutern, sondern nur die Formel für die Bestimmung von Byte 7 für Interessierte angeben:

$$\text{Byte 7} = \text{ld}\left(\frac{\text{Anzahl der Bänke} * 64\text{k}}{\text{Gesamtgröße des Bankspeichers}}\right)$$

Das Argument des dualen Logarithmus ist auf die nächste ganze Zweierpotenz aufzurunden. Beispiel:

```
Byte 7 = ld ( 8*64 / 256 ) = ld 2 = 1      beim Specci 272K
Byte 7 = ld ( 2*64 / 64 )  = ld 2 = 1      beim Specci 80 K
```

Wer z.B. acht alte 8-kByte RAMs vom Typ 6264 geerbt hat und als "unvergeßliche" 64-k-RAMDisk mit Batteriepufferung verwenden will, berechnet:

```
Byte 7 = ld ( 8*64 / 64 ) = ld 8 = 3
```

Die Parametertabelle kann wie bei LW5 mit einem BASIC-Programm mit einer 18 Elemente langen DATA-Zeile ins OPUS-RAM geschickt werden. Man beachte bei 2-Byte-Einträgen das INTEL-Format: Low-Byte zuerst! Dieses Programm kann gleichzeitig zum Formatieren der RAMDisk dienen, wenn man die Zeile 650 FORMAT 6;"RAMDISK" einfügt.

Hier ein Programm zum Testen der RAMDisk (Specci 272K). Es ist wichtig, daß die Test-Dateien, die ja immer gleich sind, eine Länge haben, die (auch nach Einteilung in Blöcke!) nicht ins 4096-/8192-/168384- usw. - Raster fällt, sonst wird bei evtl. Überschreiben von Blöcken durch einen Hardware-Fehler (wie mir einmal passiert ist) die Funktion der RAMDisk nur vorgetäuscht.

```
100 FOR i=1 TO 31
120 LET a$="tfile"+STR$ i: PRINT a$
130 SAVE *6; a$ CODE 0,7000
140 NEXT i

150 CLS: CAT 6: STOP

600 FOR i=1 TO 31
620 LET a$="tfile"+STR$ i: PRINT a$
630 VERIFY *6; a$ CODE 0
640 NEXT i
```

Wer eine optische Bank-Anzeige hat, kann verfolgen, wie bei jeder Datei zunächst kurz auf die CAT-Bank (meist Bank 1) und dann auf die höheren Bänke umgeschaltet wird.

Zum Schluß noch eine Frage an die User: Ich möchte gerne TASWORD 3 und Masterfile beim ersten Laden von Diskette auf LW6 übertragen und dann auf LW6 damit arbeiten, um die Nachladezeit zu verkürzen. Wie (POKEs?) kann ich das erreichen? Vielleicht können Kenner der Materie ihre Tips im Userclub-Info verbreiten! Übrigens bin ich ganz allgemein an "POKES" für solche Utility-Programme interessiert. — Wer hat es geschafft, HiSoft C auf OPUS Discovery zum Laufen zu bringen (incl. Speichern, Laden, Bibliothek und Drucker)? - Bitte melden!

Wer noch Fragen zur RAMDisk hat, möge mir bitte schreiben (habe meistens gerade keine Zeit zum Telefonieren...). Ich wünsche weiterhin viel Spaß beim "Specculieren" und verbleibe
Euer Claus-Jörg Kleine

Guido Schell
Auf dem Stocke 37
4972 Löhne 2

Tel. 05732 8769

Löhne, 23.04.1990

Hallo Leute !

Den Vorschlag von Rolf die Club-Kontakte gen Osten auszuweiten finde ich gut. Ich persönlich habe auch schon Briefkontakt. Der gute Mann bastelt sich einen Spectrum-Kompatiblen Rechner zusammen. Er hat die Mutterplatine aus einem Bastelladen in Erfurt. Außerdem hat er mir von einem HDT-ROM geschrieben. Das ist eine verbesserte Version des bekannten ISO-ROM. HDT bedeutet HEX-DEZ-TEXT. Ich gehe davon aus das die Verbesserungen des ISO-ROM bekannt sind. Was ist nun am HDT-ROM noch besser ? Zunächst Speichern und laden mit doppelter Geschwindigkeit (2,6 k-Baud). Der Monitor kann mit NMI oder * aufgerufen werden. Leider kenne ich den Monitor des ISO-ROM nicht. So kann ich schlecht Vergleiche anstellen. Auf jedenfall soll das HDT-ROM komfortabler als das ISO-ROM sein. Ist dieses ROM auch hier in der BRD bekannt ?
Nun zu dem Brief von Manfred Nolten im April-INFO:
Programm Konvertierung auf OPUS - ja wenn man Programme mit Nachlader hat ist das schon teuflisch. Vor allem wenn die Programme auf Cassette nicht "normal" gesaved sind. Aber wie will man sowas auseinandernehmen ? Da müßte man halbe Programme zerhacken. Ob das lohnt ? Die Idee mit der Fragebogenaktion finde ich echt gut. Es gibt hier im INFO immer wieder tolle Anregungen, aber dabei bleibt es meistens. Deshalb habe ich mir vorgenommen mal einen Fragebogen zu entwerfen. Aber das dauert wohl noch etwas. Auf jeden Fall hoffe ich auf rege Beteiligung. Oder hast Du, Manfred, schon was in der Mache ???
CARTRIDGES: Die habe ich im Oktober '89 bei Computer ACC. in München bestellt. Ende März bekam ich einen Brief das die Produktion der Cartridges eingestellt ist. Gibt es mehrere Hersteller ?
SAM COUPE: Gibt es eigentlich hier im Club jemand der schon einen COUPE hat ? Dann wäre es doch Zeit für einen Erfahrungsbericht.

Das war's dann auch schon

Spectrum User Club Wuppertal

Wilhelm Wlecke, Lönsweg 92, 4904 Enger Tel. 05224/3449

Folge 12

Enger, den 27.4.90

Hallo Spectrum User!

Diesmal gleich wieder eine Korrektur zum Beitrag im Mai-Info! Im Assemblerlisting habe ich dort in der Eile in 2 Zeilen falsche Werte angegeben und zwar muß in Zeile 420 die Zahl 10000 natürlich durch 100 und in Zeile 540 durch 10 ersetzt werden!

Die April-Beiträge fand ich wieder sehr interessant, besonders die von Herrn Sperl, Hartig, Nolten und Pokora. Die von Herrn Weiske vorgeschlagene Tastaturpufferung ist sehr wirksam und einen Kühlkörper auf dem ULA-Chip kann ich aus eigener Erfahrung auch nur empfehlen! Mit den Userport-Kontakten gibt's bei mir nur selten Probleme. Größere Sorgen bereitet mir, daß ich langsam an die Grenzen der Buss-Belastbarkeit stoße mit den vielen angeschlossenen Interfaces. Hat schon mal jemand eine

funktionierende Buss-Pufferung

aufgebaut? Das wäre doch eine tolle Sache für's Info! Zum Thema User in der DDR! Da dort der SPECTRUM weit verbreitet ist, gibt es da wirkliche Experten, sowohl was Hard- als auch Software betrifft. Vielleicht finden ja einige den Weg in den Club!

Nun aber zum heutigen Thema: Verbesserungen des an sich schon guten Assembler/Disassemblers 'SYS' Vers. 2.2! Die 1. betrifft den Disassembler 'Dismon16/ 48'! Dort gibt es die auf Seite XX beschriebene LIST-Befehl-Erweiterung. Damit kann man Code von der Ladeadr. unabhängig mit der Original-Adr. disassemblieren, falls er z.B. auf Grund von Überschneidungen an eine andere als diese geladen werden mußte. Gibt man die Routine laut Handbuch ein, so funktioniert sie leider nur bei LIST-, jedoch nicht bei DUMP. Das kann leicht geändert werden, da beide Befehle zur Adressausgabe die gleichen Routinen verwenden. Man braucht nur mit Hilfe der MONITOR-Funktion 2 Adressen ändern und zwar bei:

		Neue	alte Werte
"Dismon16":	ab h72F7 -->	hA2 / h6C	(hD6 / h72)
und	ab h7305 -->	hD4 / h6C	(hA0 / h70)
"Dismon48":	ab hF2F7 -->	hA2 / hEC	(hD6 / hF2)
und	ab hF305 -->	hD4 / hEC	(hA0 / hF0)

Jetzt wird auch bei DUMP neben Anfangs-/Endadr. eine Bezugsadr. erwartet und schon kann man sich alle MC-Programmteile mit korrekten Adressen ausgeben lassen. Es ist mir unverständlich, warum diese Verbesserung im Handbuch nicht mit angeführt wird!

Nun die 2. Verbesserung! Sie betrifft den Assembler und gilt in erster Linie für alle BETA-Disk-User. Laut Handbuch (Seite 14) wird bei BETA und TIMEX der VERIFY-Befehl ignoriert, da das DOS ihn automatisch durchführt. Also habe ich mir überlegt, daß man ihn durch einen anderen ersetzen könnte, wie CAT oder ERASE! Dazu mußte ich erst mal finden, wie und wo zwischen Normal- und Disk-Befehlen (die mit vorangestellten '*') unterschieden wird ('*CLEAR' fällt auch darunter, ist jedoch kein Disk-Befehl)!

Weil es vielleicht einige interessiert, wie man so was angehen kann, will ich versuchen meine Überlegungen und das Vorgehen möglichst genau zu beschreiben! Zuerst habe ich mir gedacht, daß es ja wohl irgendwo im MC-Programm eine Stelle geben müßte, an der auf das Symbol '*' (Code $2A) getestet wird! Da dies meist mit einem Vergleichsbefehl (compare) geschieht, ist es naheliegend, darauf hin zu untersuchen. In unserem Fall lautet der Befehl also 'CP $2A', was die Zeichenfolge '$FE / $2A' gibt. Die Sucharbeit habe ich aber dem Disassembler überlassen, dem man ja per EDIT- die Zeichenfolge übergeben und anschließend mit SEARCH-Befehl im Bereich des 'Micass'-Codes (von 56626 bis 65535) suchen lassen kann. An 4 Stellen wurde er bei meiner Version fündig und zwar bei Adr. $ECAC, $EDFC, $F744 und $F75B!

Deren Umgebung habe ich mir per DUMP näher angesehen, besonders den Bereich vor der gesuchten Zeichenfolge. Dort suche ich möglichst eine Stelle mit einem $C9 (= RET) oder $C3, $xx, $xx (= JUMP $xxxx)! Dahinter beginnt mit großer Sicherheit ein Unterprogramm, daß man mit 'LIST start,ende' disassemblieren kann. So gibt's ziemlich sicher sinnvolle Listings. Es zeigte sich, daß bei $EDFC/FD die Codefolge zu 2 verschiedenen Befehlen gehörte, also kein 'CP $2A' ergab. Um $F744 und $F75B tauchen

einige CP-Befehle auf,die hauptsächlich auf die Rechensymbole + - / * und $ % " ( ) & hin untersuchen. Auch nicht das Richtige! Der entscheidende Befehl steht bei $ECAC! Dort trennen sich die Wege, falls ein '*'-Befehl gefunden wird! Alle weiteren Einzelheiten genau zu erklären, würde zu lang, daher nur das Wesentliche in Kurzform (siehe auch Assemblerlistings):

Zuerst wird's C-Reg. mit 5 geladen als Zähler für 5 gültige *-Befehle und dieser in der Systemvar. XPTR (Adr. $5C5F =23647) gerettet (durch 'LD (IY+$25),C')! Nun wird das vorher ab $E542 abgelegte eingetippte Befehlswort mit denen in einer Tabelle ab $E7A3 verglichen (Kleinbuchstaben als große interpretiert). Wurde in der Tabelle Wortende erkannt (Code $20), ohne daß das Wort mit dem gesuchten Befehl übereinstimmte, so wird das C-Reg. um 1 vermindert und weiter verglichen. Sonst Abbruch der Suche und in C steht eine Zahl, die anzeigt, das wievielte Wort dem gesuchten entsprach. Daraus wird nun auf komplizierte Weise ein Wert errechnet, der nach Addition mit dem IX-Reg. eine Adresse ergibt. Die wird über DE auf den Stapel gelegt und mit RET angesprungen. Abhängig von der errechneten Adr. wird nun für den jeweiligen Diskbefehl eine unterschiedliche Schlüsselzahl ermittelt (außer bei '*clear'). Diese lautet bei:
***save -> $80, *load -> $81, *merge -> $82 und *verify -> $83**
und wird in $5C5F geladen. Die Schlüsselzahlen sind entscheidend für Verzweigungen zur befehlsspeziefischen Bearbeitungsroutine! Da VERIFY nur bei BETA- und TIMEX-Disk ignoriert wird, liegt der Schluß nahe, daß die weitere Behandlung dieses Befehls vom durch den File 'INSTAL.DRV' einkopierten BETA-Teil erfolgt! Sieht man sich den 1. Teil dieser Anpassungsroutine ($5E1 - $F142) an, so fällt bei aufmerksamer Durchsicht auf, daß obige Werte dort in Verbindung mit den entsprechenden Befehlscodes auftauchen:

```
$E530  $80, $F8    ->    ($F8 = Code für SAVE)
$E532  $81, $EF    ->    ($EF =    "   " LOAD)
$E534  $82, $EF    ->    ($EF =    "   " LOAD)
$E536  $83, $D6    ->    ($D6 =    "   " VERIFY)
```

Auffällig ist, daß Code $EF gleich 2 mal vorkommt. Dabei steht das 2. LOAD als Ersatz für MERGE, denn es handelt sich ja nur um ein LOAD mit neu errechneter Ladeadresse! Das BASIC-MERGE ist hier unbrauchbar, da nur für BASIC-Prog. tauglich. Aus dem doppelten Auftreten des Codes $EF kann man schließen, daß nicht der Befehlscode selbst, sondern die davor stehende Nummer zur Unterscheidung der einzelnen Befehle mit herangezogen wird!

Dies bestätigt sich auch, wenn man nachsieht, wie nach der Schlüsselzahlermittlung verzweigt wird (Adr. $EFBF-$EFC6). Dort wird erst mal geprüft, ob die Schlüsselzahl => $80 ist. Bei $80 ist es der '*savec oder *savet'-Befehl, der gesondert behandelt wird ($EFC7-$EFD7). Danach oder bei einem sonstigen Disk-Befehl, wird erst der Namenspuffer ($E639-$E648) gelöscht, der Name hineinkopiert ($EFD8-$EFDC) und nach einigen Vorbereitungen mit Hilfe der Schlüsselzahl unterschieden, ob ein Kassetten- oder Disk-Befehl bearbeitet werden soll ($EFDD-$EFE2). Da bei uns letzteres der Fall ist, wird mit dem Befehl 'JP M,$F091' in den zweiten Teil der Anpassungsroutine ($F091-$F142) gesprungen. Gleich bei $F091 steht 'CP $83' gefolgt bei $F093 von 'RET Z'! Damit ist auch klar, warum 'VERIFY' garnicht erst abgearbeitet wird, denn bei Code $83 ist das Z-Flag gesetzt und damit der Rücksprung erzwungen. Der dann folgende Programmteil generiert die bei BETA-Disk erforderliche REM-Zeile mit Befehl, Namen und CODE-Daten. Er wird bei allen anderen Disk-Befehlen durchlaufen! Zuletzt erfolgt die Ausführung durch den TRDOS-Aufruf bei Adr. $E5F9 mit anschließender Rückkehr in die Eingaberoutine!

Ersetzt man 'RET Z' durch 'NOP' (Code $00), so wird der Befehl 'VERIFY' ausgeführt, wie man schnell überprüfen kann, allerdings nicht durch Fehlermeldungen unterstützt, sodaß er wirklich unbrauchbar ist! Aber jetzt haben wir den Schlüssel für neue, sinnvollere Befehle gefunden, denn wenn VERIFY nun abgearbeitet wird, muß das bei einem 'CAT' oder 'ERASE' auch funktionieren! Dazu sind allerdings noch 2 kleine Ergänzungen notwendig:

1.) In der Tabelle von $E630 - $E638 in der die Codes $80 - $83 den Befehlscodes zugeordnet sind, müssen wir den 'VERIFY'-Code $D6 durch den gewünschten ersetzen (bei 'CAT' = $CF oder bei 'ERASE' =$D2) bei Adresse $E638 (58936)!

2.) In der Tabelle mit den erlaubten Befehlen muß natürlich 'VERIFY' mit 'CAT' oder 'ERASE' überschrieben werden:

```
Original-Version mit VERIFY
$E7A3   $43 $4C $45 $41 $52 $20    CLEAR
$E7A9   $53 $41 $56 $45 $20 $4C    SAVE L
$E7AF   $4F $41 $44 $20 $56 $45    OAD VE
$E7B5   $52 $49 $46 $59 $20 $4D    RIFY M
$E7BB   $45 $52 $47 $45 $20 $D9    ERGE .

Version mit CAT
$E7AF   $4F $41 $44 $20 $43 $41    OAD CA
$E7B5   $54 $20 $4D $45 $52 $47    T MERG
$E7BB   $45 $20 $20 $20 $20 $D9    E    .

Version mit ERASE
$E7AF   $4F $41 $44 $20 $45 $52    OAD ER
$E7B5   $41 $53 $45 $20 $4D $45    ASE ME
$E7BB   $52 $47 $45 $20 $20 $D9    RGE  .
```

Jetzt können wir den Assembler aufrufen und die neuen Befehle testen: Bei 'ERASE' also eine Disk mit zu löschenden CODE-Files einschieben, *erase "Filename" <ENTER> eintippen und schon ist der CODE-File gelöscht. Bei anderen Filetypen klappt das aus dem Assembler natürlich nicht, aber das wäre ja auch nicht sinnvoll! Ist die Wahl auf 'CAT' gefallen, tippen wir *cat <ENTER> ein... und nichts passiert! Wie das? Nach der Befehlswortprüfung wird noch auf einen Filenamen in "" getestet und genau den vermißt der Assembler nun! Also Korrektur: *cat "a:" <ENTER> (oder *cat "") und schon erscheint der Katalog von Laufwerk A. Sehen wir uns die erzeugte Zeile 5 an, so sieht sie folgendermaßen aus:

```
5 REM: CAT "a:" CODE 32768,23599
```

Das 'CODE 32678,23599' paßt eigentlich garnicht zum CAT-Befehl, wird aber hier zwangsläufig erzeugt, weil auch das alte '*verify' nicht anderes als ein abgewandelter LOAD-Befehl war! Aber glücklicherweise stört es auch nicht! Man könnte statt 'CAT' natürlich auch 'LIST' einbauen, aber da gibt's schneller Probleme mit dem Speicherplatz (Warum? Siehe meinen Artikel vom 15.7.89, Info 8/89), wenn RAMTOP auf 25000 gesetzt ist.

Wer nun glaubt, er müßte auch bei Rekorder-Betrieb auf den dort ja nicht unwichtigen VERIFY-Befehl verzichten, irrt! Allerdings muß er mit 'cat' "Filename" (oder 'erase "...") aufgerufen werden, denn 'verify "..."' wird ignoriert, weil der Befehlsname in der Tabelle ja überschrieben wurde! Aber wieso funktioniert das überhaupt? Entscheidend ist hier wieder die Schlüsselzahl (hier $03, siehe Listing Zeile 1800) denn nur sie wird bei der entscheidenden Verzweigung (bei Adr. $F0F3) abgefragt. Danach geht's dann bald in die entsprechende ROM-Routine ($0556)!

Wer neugierig geworden ist, kann das Programm ja mal selbst mittels Disassembler erforschen! Dazu eignet sich hervorragend der TRACE-Modus in Verbindung mit der Möglichkeit, Breakpoints zu setzten! Aber Achtung, das TRDOS läßt sich im TRACE-Modus nicht aufrufen, daher ist spätestens beim Befehl in Adresse $E5F9: 'CALL $3C03 (oder $3D03) Schluß zu machen! Genug für diesmal! Hoffentlich sind die Erläuterungen verständlich genug ausgefallen! Bis nächstes Mal!

```
E542          0010        ORG  $E542       ;EINGABEPUFFERBEREICH (mit Beispiel):
              0020 ;
E542          0030        DB   $2a, $5c, $6f, $61, $64, $20 ;* l o a d
E548          0040        DB   $22, $46, $69, $6c, $65, $6d ;" F i l e n
E54E          0050        DB   $61, $6d, $65, $22, $00, $00 ;a m e "
E554          0060        END

              0070 ;=============================================================
              0080 ;
E630          0090        ORG  $E630       ;TABELLE MIT SCHLÜSSELZAHLEN UND BEFEHLSCODES!
              0100 ;
E630          0110        DB   $80, $f8, $81, $ef  ;$F8 = save, $EF = load
E634          0120        DB   $82, $ef, $83, $D6  ;$EF = load (merge), $D6 = verify
E638          0130        END

              0140 ;=============================================================
              0150 ;
E7A3          0160        ORG  $E7A3       ;TABELLE MIT '*'-BEFEHLEN!
              0170 ;
E7A3          0180        DB   $43, $4c, $45, $41, $52, $20 ;C L E A R
E7A9          0190        DB   $53, $41, $56, $45, $20, $4c ;S A V E  L
E7AF          0200        DB   $4f, $41, $44, $20, $56, $45 ;O A D  V E
E7B5          0210        DB   $52, $49, $46, $59, $20, $4d ;R I F Y  M
E7BB          0220        DB   $45, $52, $47, $45, $20, $D9 ;E R G E  .
E7C1          0230        END
```

```
                0260 ;HL- zeigt auf Eingabepufferanfang ($E542) und B-Reg. enthält $2013.
                0270 ;1. Zeichen ()$20 aus Puffer ist in A, HL zeigt auf dessen Pufferadr.!
                0280 ;
ECAC            0290         org  $ECAC        ;BEGINN DER UNTERSCHEIDUNG ZWISCHEN NORMAL- UND '#'-BEFEHLEN
                0300 ;
ECAC FE2A       0310         CP   $2a          ;Ist das erste Zeichen des Befehls '#'?
ECAE 2807       0320         JR   Z,Lecb7      ;Ja? Dann weiter bei $ECB7!
ECB0 FE30       0330         CP   $30          ;Oder ist das Zeichen =) '0'?
ECB2 300D       0340         JR   NC,Lecc1     ;Dann weiter prüfen bei $ECC1!
ECB4 C3BAEB     0350         JP   $ebba        ;Weiter bei $EBBA (Wert unzulässig)!
ECB7 0E05       0360 Lecb7   LD   C,$05        ;C-Reg. mit 5 laden als Zähler für '#-Befehle)
ECB9 11A3E7     0370         LD   DE,$e7a3     ;Anfang der Tabelle der zulässigen '#'-Befehle laden.
ECBC CD3BEC     0380         CALL $ec3b        ;Nächstes Zeichen () $20 aus Puffer ins A-Reg. bringen.
ECBF 1808       0390         JR   Lecc9        ;Weiter bei $ECC9!
ECC1 FE3A       0400 Lecc1   CP   $3a          ;Ist der Code ($3A?
ECC3 DAC6F1     0410         JP   C,$f1c6      ;Dann ist es eine Zahl (0 - 9)! Weiter bei $F1C6.
ECC6 1150E7     0420         LD   DE,$e750     ;Anfangsadr. der Tabelle aller erlaubten Befehle laden!
ECC9 FD7125     0430 Lecc9   LD   (IY+$25),C   ;Stand des Befehlzählers in $5C5F laden.
                0440 ;
                0450 ;Wenn kein '#'-Befehl, dann enthält C = $13 (=19 mögliche Befehle).
                0460 ;
ECCC E5         0470 Leccc   PUSH HL           ;Puffer-Adr. des ersten Zeichens retten.
ECCD 1A         0480 Leccd   LD   A,(DE)       ;Zeichen aus Befehlswort-Tab. in A-Reg. holen und
ECCE B8         0490         CP   B            ;testen, ob es $20 (Befehlswortende) ist (B enthält auch $20)!
ECCF 281E       0500         JR   Z,Lecef      ;Dann ist Befehl gefunden! Weiter bei $ECEF!
ECD1 AE         0510         XOR  (HL)         ;Sonst durch XOR-Verknüpfung von A- und HL-Reg.-Inhalt!
                0520 ;
                0530 ;Dadurch Unterscheidung Klein-/Großschrift/sonstige Zeichen! Beispiel:
                0540 ;Oberen Zeile: das Zeichen im A-Reg. (hier $44 ='D'),
                0550 ;nächste Zeile: diverse Zeichen aus HL (=Eingabepuffer),
                0560 ;untere Zeile: Ergebnis in A-Reg. (ungleiche Bits ergeben eine 1).
                0570 ;
                0580 ;'D'=$44 = 01000100    'D'=$44 = 01000100    'D'=$44 = 01000100
                0590 ;'D'=$44 = 01000100    'd'=$64 = 01100100    '.'=$2E = 00101110
                0600 ;          --------          --------          --------
                0610 ;    $00 = 00000000      $20 = 00100000      =$6A = 01101010
                0620 ;
ECD2 2817       0630         JR   Z,Leceb      ;Korrekte Großschrift? Dann Z-Flag gesetzt, weiter ) $ECEB!
ECD4 B8         0640         CP   B            ;Oder steht im A-Reg. nun $20 (wie in B!)?
ECD5 2814       0650         JR   Z,Leceb      ;Dann korrekte Kleinschrift! Zum nächsten Zeichenvergleich!
ECD7 7E         0660         LD   A,(HL)       ;Keins davon? Dann ursprünglichen Code aus Puffer holen
ECD8 B8         0670         CP   B            ;und prüfen, ob es ein Code $20 (Leerzeichen) ist!
ECD9 2815       0680         JR   Z,Lecf0      ;Dann weitermachen bei $ECF0!
ECDB FE2E       0690         CP   $2e          ;oder ist es ein '.'? (abgekürzter Befehl)
ECDD 2811       0700         JR   Z,Lecf0      ;Dann auch weiter bei $ECF0, da Vergleich beendet!
ECDF 1A         0710 Lecdf   LD   A,(DE)       ;Sonst nächstes Zeichen aus Tabelle holen,
ECE0 13         0720         INC  DE           ;DE-Reg. auf nächste Tabellenadr. erhöhen
ECE1 B8         0730         CP   B            ;und prüfen, ob das Zeichen Code $20 ist (Wortende).
ECE2 20FB       0740         JR   NZ,Lecdf     ;Nein? Dann zurück und weiter danach suchen!
ECE4 E1         0750         POP  HL           ;Wenn Wortende gefunden, Pufferanfangsadr. vom Stapel holen!
ECE5 0D         0760         DEC  C            ;Befehlswortzähler um 1 erniedrigen
ECE6 20E4       0770         JR   NZ,Leccc     ;und falls () 0, nächsten Befehl mit Eingabe vergleichen!
ECE8 C3B9EB     0780         JP   $ebb9        ;Sonst Abbruch, da Eingabe falsch!
                0790 ;
ECEB 13         0800 Leceb   INC  DE           ;Tabellenadresse
ECEC 23         0810         INC  HL           ;Eingabepufferadresse jeweils um 1 erhöhen!
ECED 18DE       0820         JR   Leccd        ;und weiter zum nächsten Zeichenvergleich!
                0830 ;
ECEF 2B         0840 Lecef   DEC  HL           ;HL auf vorletztes Zeichen im Eingabepuffer zurücksetzen.
ECF0 DD2114ED   0850 Lecf0   LD   IX,$ed14     ;IX mit Basisadr. für Sprungadr.-Berechnung laden und
ECF4 0600       0860         LD   B,$00        ;B mit 0 (beides wird nur bei Nicht-'#'-Befehlen gebraucht!.
ECF6 FDCB2556   0870         BIT  2,(IY+$25)   ;Bit 2 von $5C5F ist nur bei Nicht-'#'-Befehlen gesetzt
ECFA 2806       0880         JR   Z,Led02      ;Bei denen ist Z-Flag auch gesetzt! Dann weiter bei $ED02!
ECFC DD213A6D   0890         LD   IX,$6d3a     ;Bei '#'-Befehl $6D3A als Basis für Sprungadr.-Berechnung
ED00 0680       0900         LD   B,$80        ;ins IX- und $80 ins B-Reg. laden!
                0910 ;
                0920 ;Im B- steht nun $80, im C-Reg. die Nummer des eingegebenen Befehls,
                0930 ;d.h. bei #clear = 5, #save = 4, #load = 3, #verify = 2, #merge = 1!
                0940 ;
ED02 D9         0950 Led02   EXX               ;Erst- und Zweit-Registersatz vertauschen!
ED03 68         0960         LD   L,B          ;B- nach L-Reg. umladen
ED04 D9         0970         EXX               ;und Registersätze wieder vertauschen!
ED05 CB21       0980         SLA  C            ;Nun C-Reg.-Inhalt verdoppeln durch Linksschieben der Bits in C!
                0990 ;
                1000 ;Z.B. bei C-Inhalt 3 =) 00000011  -) nach SLA C -) 00000110 = 6
                1010 ;
ED07 DD09       1020         ADD  IX,BC        ;Nun hier IX + BC, Ergebnis in IX = Adresszeiger
                1030 ;
                1040 ;Bei #clear : IX + BC = $6D3A + $800A = $ED44 -) zeigt auf Adr. $EED5
                1050 ;  "  #save  :   "    = $6D3A + $8008 = $ED42 -)  "   "  "  $EF3F
                1060 ;  "  #load  :   "    = $6D3A + $8006 = $ED40 -)  "   "  "  $EF3E
                1070 ;  "  #verify:   "    = $6D3A + $8004 = $ED3E -)  "   "  "  $EF3C
                1080 ;  "  #merge :   "    = $6D3A + $8002 = $ED3C -)  "   "  "  $EF3D
                1090 ;
ED09 D1         1100         POP  DE           ;Alten Wert von DE-Reg. vom Stapel nehmen, da nun überflüssig!
ED0A 118CEC     1110         LD   DE,$ec8c     ;$EC8C als Rücksprungadr. in Eingabeschleife laden
ED0D D5         1120         PUSH DE           ;und für später retten!
ED0E DD5E00     1130         LD   E,(IX+$00)   ;Nun Wert der Adr. auf die IX zeigt in DE laden.
ED11 DD5601     1140         LD   D,(IX+$01)   ;DE enthält damit für jeden Befehl eine andere Adresse!
ED14 D5         1150 Led14   PUSH DE           ;Diese auf dem Stapel ablegen und
ED15 C9         1160         RET               ;gleich wieder vom Stapel nehmen und anspringen.
ED16            1170         END
```

```
EB3C          1200          ORG  $eb3c       ;TABELLE MIT SPRUNGADRESSEN ZUR SCHLÜSSELZAHLERMITTLUNG!
              1210 ;
EB3C          1220          DB   $bd, $ef     ;Diese Adr. ($EFBD) wird bei 'merge' angesprungen!
EB3E          1230          DB   $bc, $ef     ;   "   "   ($EFBC)   "   "  'verify'    "
EB40          1240          DB   $be, $ef     ;   "   "   ($EFBE)   "   "  'load'      "
EB42          1250          DB   $bf, $ef     ;   "   "   ($EFBF)   "   "  'save'      "
EB44          1260          DB   $d5, $ee     ;   "   "   ($EED5)   "   "  'clear'     "
EB46          1270          END

              1280 ;=================================================================
              1290 ;
EFBC          1300          ORG  $EFBC        ;SCHLÜSSELZAHLERMITTLUNG UND VERZWEIGUNG ZU '#'-BEFEHLEN!
              1310 ;
EFBC 04       1320          INC  B            ;Hierhin erfolgt Sprung bei 'verify'!
EFBD 04       1330          INC  B            ;   "      "     "    "  'merge'!
EFBE 04       1340          INC  B            ;   "      "     "    "  'load'!
EFBF FD7025   1350          LD   (IY+$25),B   ;   "      "     "    "  'save'!
              1360 ;
              1370 ;Schlüsselzahl in B ist nur bei save unverändert $80, bei load $81,
              1380 ;merge $82 und verify $83! Wert in (IY+$25) = $5C5F ablegen!
              1390 ;
EFC2 3E7F     1400          LD   A,$7f        ;A-Reg. mit 127 (binär 01111111) laden und damit Wert aus
EFC4 A0       1410          AND  B            ;B maskieren. Nur bei B = $80 (10000000) ist das Ergebnis 0!
              1420 ;
              1430 ;AND-Verknüpfung: Beide Bits 1? Dann Ergebnis ebenso 1! Sonst 0!
              1440 ;Verknüpfungsergebnis landet in A! Nur bei A = 0 ist Z-Flag gesetzt!
              1450 ;
              1460 ;$7F = 01111111    $7F = 01111111    $7F = 01111111    $7F = 01111111
              1470 ;$80 = 10000000    $81 = 10000001    $82 = 10000010    $83 = 10000011
              1480 ;      --------          --------          --------          --------
              1490 ;$00 = 00000000    $01 = 00000001    $02 = 00000010    $03 = 00000011
              1500 ;
EFC5 2011     1510          JR   NZ,Lefd8     ;Nicht gesetzt? Also kein save-Befehl! Weiter bei $EFD8!
EFC7 CD3BEC   1520          CALL $ec3b        ;Doch? Dann nächstes Zeichen ()$20 von Puffer in A laden
EFCA F620     1530          OR   $20          ;und alle Buchstaben wie kleine behandeln!
              1540 ;
              1550 ;OR-Verknüpfung: Eins oder beide Bits 1? Dann Ergebnis 1! Sonst 0!
              1560 ;Verknüpfungsergebnis landet in A! Nur bei A = 0 ist Z-Flag gesetzt!
              1570 ;
              1580 ;$20 = ' ' = 00100000    $20 = ' ' = 00100000    $20 = ' ' = 00100000
              1590 ;$43 = 'C' = 01000011    $54 = 'T' = 01010100    $63 = 't'= 01110100
              1600 ;           --------          --------          --------
              1610 ;$63 = 'c' = 01100011    $74 = 't' = 01110100    $63 = 't = 01110100
              1620 ;
EFCC FE63     1630          CP   $63          ;Ist es 'c'? Dann wurde 'savec' eingegeben und
EFCE 2805     1640          JR   Z,Lefd5      ;es soll Objektcode gespeichert werden! Weiter bei $EFD5!
EFD0 FE74     1650          CP   $74          ;Oder ist es 't'? Dann wurde 'savet' eingegeben (Quelltext)!
EFD2 C2B9EB   1660          JP   NZ,$ebb9     ;Keins von beiden? Dann Eingabe nicht korrekt, Abbruch!
EFD5 32605C   1670 Lefd5    LD   ($5c60),A    ;Sonst den Code von 'c' oder 't' in $5C60 retten!
EFD8 C5       1680 Lefd8    PUSH BC           ;BC-Reg. retten (B enthält die errechnete Schlüsselzahl!)
EFD9 CD37EB   1690          CALL $eb37        ;Puffer löschen (16 Zeichen ab $E639). Namen hineinkopieren
              1700 ;
              1710 ;Dabei wird in $5C5B (=IY+$21) Anzahl der nicht genutzten Zeichen des
              1720 ;Namens (max. 12, z.B.: 'a:Filename') abgelegt!
              1730 ;
EFDC C1       1740          POP  BC           ;BC-Reg. zurückholen!
EFDD 3EFF     1750          LD   A,$ff        ;A-Reg. mit $FF laden und mit Wert aus B-Reg.
EFDF A0       1760          AND  B            ;AND-Verknüpfung vornehmen.
              1770 ;
              1780 ;Hierbei steht im A- anschließend der gleiche Wert wie im B-Reg.! Bei
              1790 ;AND wird auch das S-Flag gesetzt, wenn das Ergebnis =)$80 ist!
              1800 ;
EFE0 FA91F0   1810          JP   M,$f091      ; S-Flag gesetzt? Dann war's ein #-Befehl. Weiter bei $F091
              1820 ;
              1830 ;Ab hier werden Kassettenbefehle weiterbehandelt! Schlüsselzahlen
              1840 ;für Kassettenbefehle: SAVE =$00, LOAD =$01, MERGE =$02, VERIFY =$03!
EFE3          1850          END

              1860 ;=================================================================
              1870 ;
F091          1880          ORG  $F091        ;'#VERIFY'-ABWEISUNG
              1890 ;
F091 FE83     1900          CP   $83          ;Ist die ermittelte Schlüsselzahl $83?
F093 C8       1910          RET  Z            ;Dann Abbruch der Bearbeitung und Rückkehr in Eingaberoutine!
              1920 ;
              1930 ;HIER BEGINNT FÜR ALLE ÜBRIGEN DISK-BEFEHLE DIE REM-ZEILENERZEUGUNG!
              1940 ;
F094 2A5F5C   1950          LD   HL,($5c5f)   ;Schlüsselzahl (und bei save Code 't' bzw 'c') in HL
F097 22765C   1960          LD   ($5c76),HL   ;und dann in Systemvariable $5C76 ablegen!
F09A ED7B3D5C 1970          LD   SP,($5c3d)   ;Stapelzeiger auf Adr. aus ERR SP setzen und
F09E 210313   1980          LD   HL,$1303     ;Adr. ROM-Rout. 'Rückkehr nach Zeilenausführung' laden und
F0A1 E3       1990          EX   (SP),HL      ;als Rücksprungadr. nach REM-Zeilenausführung auf Stapel!
F0A2 FD3600FF 2000          LD   (IY+$00),$ff ;ERR NR auf $FF (= Wert für 'kein Fehler') setzen!
F0A6 216F15   2010          LD   HL,$156f     ;Nun einen Teil der ROM-Routine "Zeile in Progr. einfügen"
F0A9 1142E5   2020          LD   DE,$e542     ;(von $156F-$15AA) in Eingabepuffer ab $E542 kopieren.
F0AC 013C00   2030          LD   BC,$003c     ;(Routinenlänge = 60 Zeichen)
F0AF EDB0     2040          LDIR
F0B1 3EC9     2050          LD   A,$c9        ;Code für RET
F0B3 12       2060          LD   (DE),A       ;ans Ende der Kopie schreiben.
F0B4 3E1D     2070          LD   A,$1d        ;Nun max. Länge der zu erzeugenden Zeile festlegen (=29) und
F0B6 FD9621   2080          SUB  (IY+$21)     ;davon Anzahl der nicht genutzten Namenszeichen abziehen.
F0B9 6F       2090          LD   L,A          ;Das Ergebnis (= tatsächliche Zeilenlänge) in L bringen
```

```
F0BA 60        2100          LD   H,B          ; und H-Reg. löschen!
F0BB 0E05      2110          LD   C,$05        ;C-Reg. mit der vorgesehenen Zeilennummer laden und
F0BD ED43495C  2120          LD   ($5c49),BC   ;in E PPC (Nummer der laufenden Prog.-Zeile) ablegen.
F0C1 E5        2130          PUSH HL           ;HL-Reg. (Zeilenlänge!) und
F0C2 C5        2140          PUSH BC           ;BC-Reg. (Zeilennummer) retten!
F0C3 CD42E5    2150          CALL $e542        ;ROM-Rout.-Kopie aufrufen und Platz für Zeile 5 schaffen!
F0C6 E1        2160          POP  HL           ;Nun die Zeilennr. vom Stapel in HL holen und dann ROM-Rout.
F0C7 CD6E19    2170          CALL $196e        ;'Zeilenanfang suchen' aufrufen, Nummer und Länge einfügen!
F0CA 23        2180          INC  HL           ;HL-Reg. zeigt auf Zeilenanfang, deshalb durch viermaliges
F0CB 23        2190          INC  HL           ;hochzählen Zeilennummer (2 Stellen) und
F0CC 23        2200          INC  HL           ;Zeilenlängenangabe (2 Stellen) überspringen.
F0CD 23        2210          INC  HL           ;HL-Reg. zeigt danach auf Adr. für's Befehlswort!
F0CE 225D5C    2220          LD   ($5c5d),HL   ;Diesen Wert in CH ADD (=nächstes zu interpr. Zeichen).
F0D1 36EA      2230          LD   (HL),$ea     ;In diese Adr. den Befehlswort-Code von 'REM' schreiben,
F0D3 23        2240          INC  HL           ;HL eins weiter,
F0D4 363A      2250          LD   (HL),$3a     ;den ":" einfügen und
F0D6 23        2260          INC  HL           ;HL-Reg. nochmals erhöhen!
F0D7 E5        2270          PUSH HL           ;Diese aktuelle Zeilen-Adresse erst mal auf Stapel retten!
F0D8 3A765C    2280          LD   A,($5c76)    ;Nun Schlüsselzahl ins A-Reg. holen
F0DB 4F        2290          LD   C,A          ;und nach C umladen.
F0DC 2130E6    2300          LD   HL,$e630     ;Nun Anfangsadr. der Schlüsselzahlen-/ Befehlscode-Tabelle
F0DF EDB1      2310          CPIR              ;laden und diese auf Code im A-Reg. (=Schlüsselzahl) durchsuchen!
F0E1 7E        2320          LD   A,(HL)       ;HL zeigt nun auf zugehörigen Befehlscode, diesen ins A-Reg.
F0E2 E1        2330          POP  HL           ;Aktuelle Zeilenadr. vom Stapel holen und
F0E3 77        2340          LD   (HL),A       ;den Befehls-Code in Zeile einfügen ($F8, $EF oder $D6)!
F0E4 08        2350          EX   AF,AF'       ;Befehls-Code erst mal im Alternativ-Reg. zwischenspeichern
F0E5 23        2360          INC  HL           ; und HL auf nächste Zeilenadr. setzen.
F0E6 3622      2370          LD   (HL),$22     ;Das Anführungszeichen einfügen,
F0E8 23        2380          INC  HL           ;HL wieder erhöhen,
F0E9 C1        2390          POP  BC           ;tatsächliche Zeilenlänge nun ins BC-Reg. holen
F0EA 79        2400          LD   A,C          ;und ins A-Reg. umladen! Davon den Wert für min. Zeilenlänge
F0EB D612      2410          SUB  $12          ;abziehen (=18 = Zeilenlänge mit "" als Namen!.
F0ED 4F        2420          LD   C,A          ;Ergebnis (=Namenslänge) in C zurückschreiben!
F0EE 2807      2430          JR   Z,Lf0f7      ;War es 0? Dann kein Name (= "")! Weiter bei $F0F7!
F0F0 1139E6    2440          LD   DE,$e639     ;Sonst Anfangsadr. des Programmnamen-Puffers in DE laden!
F0F3 EB        2450          EX   DE,HL        ;Reg.-Tausch vornehmen und den Namen anschließend in die
F0F4 EDB0      2460          LDIR              ;Zeile umkopieren!
F0F6 EB        2470          EX   DE,HL        ;Erneut Reg.-Tausch. HL zeigt  nun auf Adr. hinterm Namen!
F0F7 3622      2480 Lf0f7    LD   (HL),$22     ;An diese Adr. den Code von "" schreiben
F0F9 23        2490          INC  HL           ;und HL auf nächste Adr. einstellen!
F0FA 36AF      2500          LD   (HL),$af     ;Dahin den Befehl 'CODE' (=$AF) schreiben.
F0FC E5        2510          PUSH HL           ;Dann aktuelle Zeilenadr. erst mal retten
F0FD 08        2520          EX   AF,AF'       ;und Befehlscode aus Alternativreg. zurückholen!
F0FE FEF8      2530          CP   $f8          ;War es ein 'save'-Befehl?
F100 381D      2540          JR   C,Lf11f      ;Nein? Dann weiter bei $F11F!
F102 3A775C    2550          LD   A,($5c77)    ;Sonst hier abgelegten Code holen ("c" oder "t") und
F105 0F        2560          RRCA              ;'Rechts durch C-Flag rotieren', also Bit 0 ins C-Flag! Bei "c"
F106 3809      2570          JR   C,Lf111      ;(=$63 =01100011) nach $F111 zur Adr.- u. Längenberechnung,
F108 CD43F1    2580          CALL Lf143        ;bei "t" (=$74 =01110100) dagegen nach $F143!
F10B 22E6E5    2590          LD   ($e5e6),HL   ;Ermittelte Quelltext-Anfangsadr. hier ablegen!
F10E EB        2600          EX   DE,HL        ;Vertauschen! Nun HL =Quelltext-Länge, DE =Quelltext-Anfang.
F10F 182D      2610          JR   Lf13e        ;Ermittelte Werte in Zeile einbauen und Zeile aufrufen!
F111           2620          END

               2630 ;
               2640 ;=============================================================================
               2650 ;
F110           2660          ORG  $F110        ;STARTADR. UND LÄNGENBERECHNUNG BEI 'SAVEC, LOAD, MERGE'!
               2670 ;
F110 2B        2680          DEC  L
F111 2A54FF    2690 Lf111    LD   HL,($ff54)   ;Bei 'savec': Anfang des Objektcodepuffers in HL
F114 22E6E5    2700          LD   ($e5e6),HL   ;und in $E5E6 ablegen!
F117 EB        2710          EX   DE,HL        ;Durch Reg.-Tausch Adr. des Objektcode-Anfangs in DE bringen
F118 2A69E6    2720          LD   HL,($e669)   ;und HL mit dessen Endadresse laden!
F11B ED52      2730          SBC  HL,DE        ;Jetzt aus beiden Werten die Objektcode-Länge berechnen
F11D 181E      2740          JR   Lf13d        ;und zum Einbau der Adr. in die Zeile und Ausführung!
               2750 ;
F11F CD43F1    2760 Lf11f    CALL Lf143        ;Bei load/merge: erst Quelltext-Anfang/Ende berechnen!
F122 FDCB3C46  2770          BIT  0,(IY+$3c)   ;Test ob Bit 0 von Schlüsselzahl in $5C76 =0! Nicht 0?
F126 2005      2780          JR   NZ,Lf12d     ;Dann war Bit 0 =1, da Schlüsselzahl bei 'load' = $81!
               2790 ;
F128 19        2800          ADD  HL,DE        ;Sonst 'merge'! Dann Anfang+Länge = Quelltext-Ende errechnen
F129 2B        2810          DEC  HL           ;und davon 1 abziehen, da Ende vorher auf Ende-Marker zeigte!
F12A 327FE6    2820          LD   ($e67f),A    ;Code des Ende-Markers (=$FF) in $E67F ablegen!
               2830 ;
F12D 227DE6    2840 Lf12d    LD   ($e67d),HL   ;Bei 'load': Anfangsadr. des Quelltextes, bzw. bei
F130 22E6E5    2850          LD   ($e5e6),HL   ;'merge' Endadr. = Ladeadresse hier 2# ablegen!
F133 EB        2860          EX   DE,HL        ;Durch Reg.-Tausch die Ladeadresse in DE bringen
F134 2A63E6    2870          LD   HL,($e663)   ;und HL mit der Anfangsadr. des Assemblers laden!
F137 25        2880          DEC  H            ;Davon 256 abziehen, um etwas Platz zu behalten.
F138 ED52      2890          SBC  HL,DE        ;Aus dieser Adr. - Ladeadr. den restlichen Platz errechnen!
F13A 3001      2900          JR   NC,Lf13d     ;Ist noch Platz für Quelltext? Dann Zeile fertigstellen!
F13C 24        2910          INC  H            ;Sonst steht in H-Reg. $FF, auf 0 setzen! Nun Wert in HL <$FF!
F13D 23        2920 Lf13d    INC  HL           ;Zur Verfügung stehender Platz + 1!
F13E C3E1E5    2930 Lf13e    JP   $e5e1        ;Ladeadr. und Länge in Zeile einbauen und ausführen!
F141 00        2940          NOP
F142 00        2950          NOP
F143 2A50FF    2960 Lf143    LD   HL,($ff50)   ;Anfangsadr. des Quelltextpuffers in HL holen
```

```
F149 4F      3000        LD   C,A        ;und C-Register!
F14A 3D      3010        DEC  A          ;0 - 1 = 255 ($FF) = Code für Textende-Marker!
F14B EDB1    3020        CPIR            ;Diesen im Quelltext suchen. Gefunden? Dessen Adr. dann in HL!
F14D D1      3030        POP  DE         ;Nun die Adr. des Quelltext-Anfangs vom Stapel in DE holen
F14E ED52    3040        SBC  HL,DE      ;und Ende-Marker-Adr. - Anfang = Länge! Ergebnis in HL.
F150 EB      3050        EX   DE,HL      ;Nach Tausch: DE = Quelltext-Länge, HL = Quelltext-Anfang!
F151 C9      3060        RET             ;Fertig! Deshalb zurück nach $F10B oder $F122!
F152         3070        END

             3080 ;
             3090 ;=================================================================
             3100 ;
E5E1         3110        ORG  $E5E1      ;STARTADR. UND LÄNGE IN ZEILE EINBAUEN UND DIESE AUSFÜHREN!
             3120 ;
E5E1 22EEE5  3130 Le5e1  LD   ($e5ee),HL ;Code- oder Text-Länge in $E5EE ablegen.
E5E4 E1      3140        POP  HL         ;Nun aktuelle Zeilenadr. wieder vom Stapel holen und in DE-Reg.
E5E5 110080  3150        LD   DE,$8000   ;  Quelltext- (32768) oder Objektcode-Anfang (25000) laden
E5E8 CD0FE6  3160        CALL $e60f      ;Jeweilige Zahl in die Zeile einbauen!
E5EB 362C    3170        LD   (HL),$2c   ;Ein Komma dahinter setzen!
E5ED 112F5C  3180        LD   DE,$5c2f   ;Eben hier abgelegte Code-/Text-Länge (z.B.23599) in DE!
E5F0 CD0FE6  3190        CALL $e60f      ;Auch diese Angabe in die Zeile einfügen!
E5F3 360D    3200        LD   (HL),$0d   ;Als Zeilenende den 'ENTER'-Code = $0D dahinter setzen!
E5F5 215827  3210        LD   HL,$2758   ;HL mit 10072 laden! Sinn ist mir z.Z. noch nicht klar!!!
E5F8 D9      3220        EXX             ;Alle Reg. mit dem Alternativ-Reg.-Satz vertauschen
E5F9 CD033D  3230        CALL $3d03      ;und TRDOS Vers.5.xx aufrufen (bei Vers. 4.xx =$3C03)!
E5FC C369EC  3240        JP   $ec69      ;Nach ordnungsgemäßem Beenden zurück in die Eingaberoutine!
E5FF         3250        END
```

29.4.90

Nun noch ein kleiner Nachtrag, obwohl der Beitrag schon länger ist, als ursprünglich geplant! Wer nämlich genau hingesehen hat, der wird sich vielleicht fragen, wieso denn überhaupt die von der Routine ab Zeile 1950 gebildete REM-Zeile abgearbeitet werden kann? Denn normalerweise müßte sie ja in der korrekten BETA-Syntax (hier für Vers. 5.03) z.B. so aussehen:

```
5 RANDOMIZE USR 15619: REM: LOAD "Filename" CODE 32768,23599
```

Der RANDOMIZE-Aufruf fehlt jedoch in der gebildeten Zeile! Das der Aufruf trotzdem funktioniert liegt daran, daß wir in Zeile 2220 (siehe Assemblerlisting) die Adresse des REM-Befehl in die Systemvariable CH ADD ($5C5D) gerettet haben. Wird jetzt mit CALL $3D03 (=15619) in Zeile 3230 das TRDOS aufgerufen, so wird nämlich auf diese Variable CH ADD zugegriffen, denn sie zeigt ja auf das nächste zu interpretierende Zeichen. Ab da wird dann zuerst das 'REM' gesucht, alles auf richtige Syntax getestet und falls Fehlerfrei, der Befehl ausgeführt! Im Grunde ist hier nur die gleiche Situation geschaffen worden, wie sie beim Aufruf der vollständigen Zeile nach Ausführung des 'RANDOMIZE USR 15619' herrschen würde, denn auch dann zeigt CH ADD auf das Zeichen ':' dahinter und der Rest läuft wie gehabt!
Diesmal war mein Beitrag nun sehr speziell, aber ich hoffe, daß auch der eine oder andere Maschinensprache-Interessierte, jedoch 'Nicht'-BETA-Disk oder 'SYS'-Besitzer, etwas damit anfangen kann! Deshalb auch die ausführlichen Kommentare! Also nochmals, bis demnächst!

Liebe User !

Hiermit möchte ich mich für die Zuschriften zu meinem Hilferuf aus INFO 3/90 bedanken.
Inzwischen bin ich glücklicher Besitzer eines 24 Nadlers (**star LC 24-10**) und mit diesem völlig zufrieden.
Alle Programme arbeiten mit ihm genauso fehlerfrei zusammen, wie mit dem *alten* 9-Nadler CENTRONICS GLP II (nur eben mit viel besserem Schriftbild in LQ statt NLQ.)
Sogar mein altes Bildschirm-COPY-Programm läuft tadellos, weil der Drucker hier seine 'überflüssigen' Nadeln abgeschaltet hat.
Im Moment schreibe ich an einem 24 Nadeln benutzendem COPY-Programm, das ganz fein aufgelöst, (aber leider auch recht winzig) SREEN-Copies druckt.

Auf alle Fälle kann ich die Konfiguration SPECTRUM-DICOVERY + **star LC 24-10** nur jedem, der an Druckerkauf denkt, empfehlen (um soviel teurer sind die paar Nadeln gar nicht).

Jürgen Schomburg
Dr.-Martin-Luther-Str.9
8580 Bayreuth

HALLO FREUNDE

Mein Nachbar ist Buchdrucker, versteht also einiges von Typographie. Ich habe ihm das DTP gezeigt. Jetzt kauft er sich einen Specci. Bisher hat er den kleinen schwarzen Kasten wie viele vor ihm- maßlos unterschätzt. Sein PC steht schon unter "Verstaube ....". Jetzt wißt Ihr, woher die Idee stammt, graphischen Schmuck zusammenzutragen, um Bausteine und Anregungen für eine ansprechende Seitengestaltung nicht erst umständlich suchen zu müssen, wenn man sie braucht. Warum muß es immer nur trockener Text sein, warum nur kaltes Glas und nackter Beton, liebe Architekten? Der Specci bietet eine überraschende Vielfalt an graphischen Möglichkeiten. Natürlich hat der kleine Sinclair seine Grenzen, aber eine große Unix-Anlage hat sie auch - bittärrscheen!
Also Freunde, macht erst mal das, was innerhalb dieser Grenzen liegt. Message understood ?
Heute möchte ich Euch den ersten Teil einer typographischen Reihe vorstellen:

DECOR PACKS

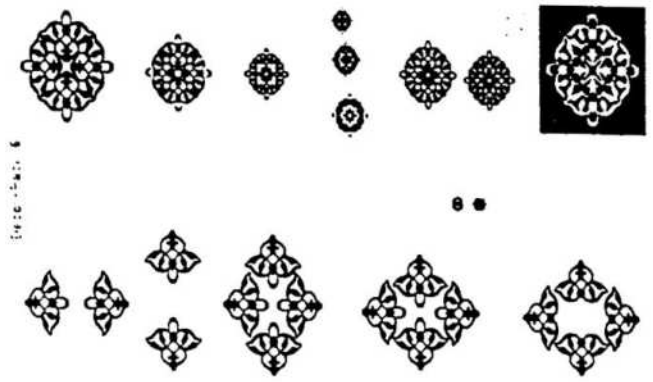
In loser Folge kommen dann die anderen über das Info. Sommerflaute ist 1990 nicht angesagt. Vorbereitet sind:

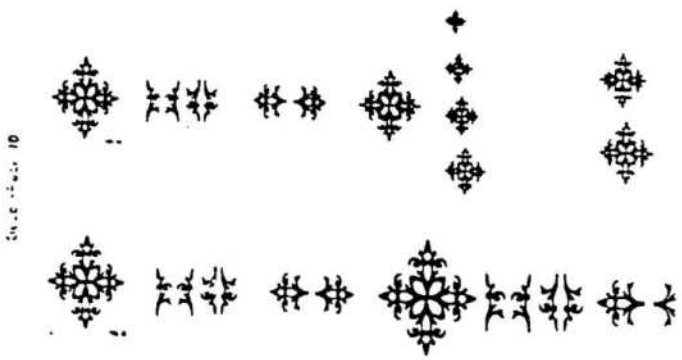
BORDER-, CORNER-, SHADE-PACKS

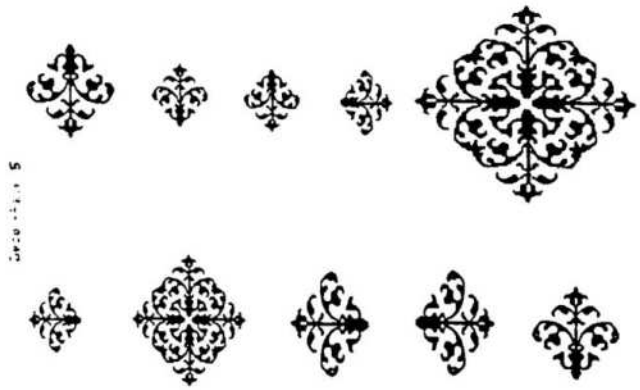
und natürlich Schriften, Initialen, sowie Anwenderbeispiele (siehe "Plakat").
All die Dinge könnt Ihr gerne von mir haben, wenn Ihr mir eine Kassette (TD oder MD) verwechslungssicher mit Eurer Anschrift und Eurem Wunsch versehen nebst einem Label für die Rücksendung sowie sorry, leidiges Thema- einem Unkostenbeitrag von DM 3.- je Pack oder Schrift in Scheinen portofrei zuschickt. Ich will nicht an Freunden verdienen, aber ans Zulegen habe ich auch nicht gedacht; vorher gebe ich mir eine Mozartkugel !
Einmal im Monat kann ich Überstunden in Freizeit umwandeln und so einen Kopiertag einschalten. Bitte um Geduld statt einer wöchentlichen Anfrage, wann endlich . . .; zum Briefe schreiben habe ich ein ähnliches Verhältnis wie der Deibl zum Weihwasser. So, und nun sitze ich doch vor dem Aquarium und schau den Pixeln zu.

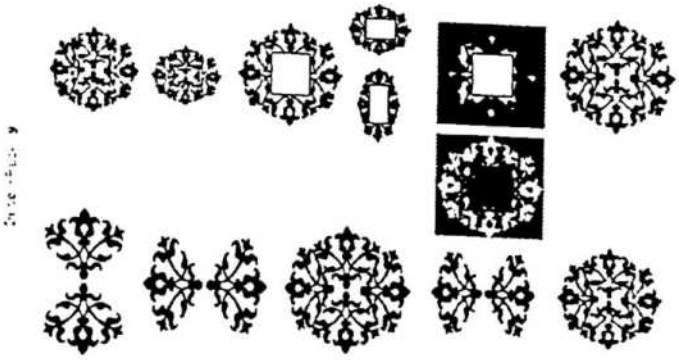
Grüße! Euer Walter Sperl Uferstraße 108, A-2525 Schwarzau

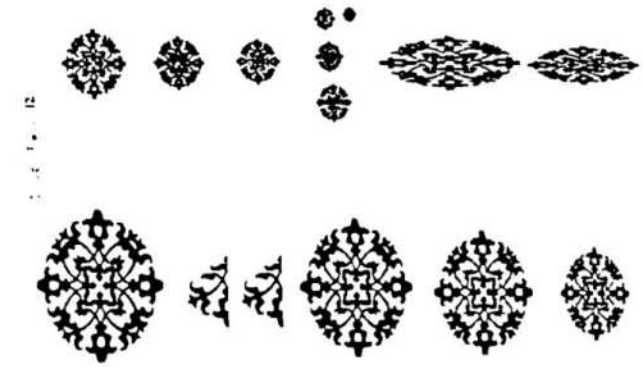
PS: Bitte entschuldigt die Fehler im Beitrag zum Info:
a) "Plakat" vergessen mitzuschicken
b) Tapfehler (bei "Tippfehler" ist doch alles ok, oder?)
c) Bildunterschriften vertauscht.
Noch eine Warnung an das Druckfehler-Teufelchen: Wenn Du nochmals zuschlägst, werde ich Dir mit der

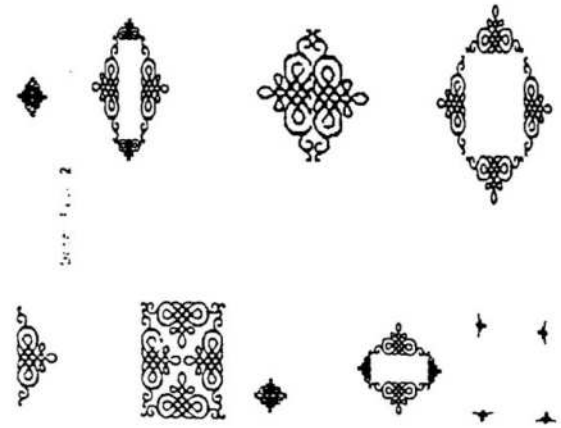


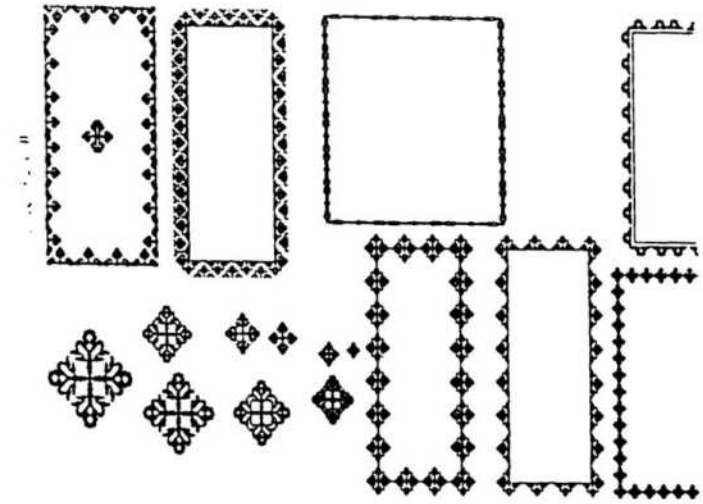


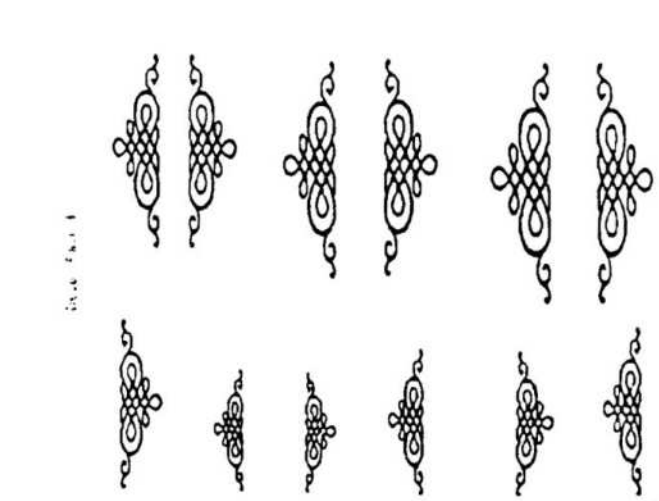












==========
DOCUMENTATION FÜR DEN RAM-IMAGE-SPLITTER V.3.1 (BETA V.4.xx)
==========

==========
= Bernhard Lutz =
= Hammerstr. 35 =
= D-6729 Bellheim =
= West-Germany =
= Tel. 07272 71444 =
==========

WAS DER SPLITTER FÜR DICH MACHEN KANN:

Er kann RAM-IMAGES (Magic Button Files) auf Disk oder Tape oder Kassette kopieren (z.B. via Disk-Master) ! Es ersetzt mit Hilfe der Beta-Disk Magic Button-Funktion Interfaces wie z.B. Multiface etc.
Er speichert das IMAGE mit oder ohne Screen und zerteilt den Hauptcodeblock in zwei Teile einer aufwärts von Adresse 27296 und den zweiten von 23296 bis 65535, welcher später in den Screen geladen wird.
Diese Blöcke können dann einfachst auf Tape, Disk oder andere Speichermedien gebracht werden (auch auf Microdrive).Oder per serieller Schnittstelle sogar zu anderen Computern. Man braucht von nun ein keine Programme mehr zu cracken, um Pokes zu suchen, oder wenn man in den Programmen etwas verändern will.

Wie man das Programm auf der Disk installiert:

Man formatiert eine SINGLE SIDED Disk und macht Platz für das RAM-Image darauf mit folgendem Befehl (wenn man im DOS ist):
SAVE "IMAGE"CODE 16384,49152.
Das schafft Platz auf der Disk für RAM Image, das ja immer ab Track 1 Sector 0 abgelegt wird und schützt später das Splitter Programm vor dem Überschreiben, wenn der Magic Button benutzt wird !
Dann wird das Splitter Programm eingegeben, und mit dem Namen "boot" auf die Single Sided Disk abgesaved. (Das sorgt dafür, das es nach einem RESET automatisch geladen ("gebootet") wird.

Wie man den Splitter benutzt: (v3.1):

1. Ein Programm deiner Wahl laden (16 or 48k)
2. Den MAGIC- (NMI-) Knopf drücken wenn das Programm einen Augenblick wartet (um z.B. die Highscores anzuzeigen oder wenn GAME OVER auf dem Screen steht.) oder wenn das Programm auf den Druck einer Taste wartet um fortzufahren.
3. Nun den RESET-Button drücken damit das "boot"-program (SPLITTER) automatisch geladen ("gebootet") wird.

4. Nun kann man WAHLWEISE mit Option 7 testen ob das gesplittete Programm einwandfrei läuft.
   Sollte es nicht so sein, dann bitte Punkte a-e beachten !!
5. Mit den Optionen SAVE+COPY 16/48k oder SCREEN kann nun das gewünschte gesaved werden:
6. Jetzt einen maximal 7 Zeichen langen File-Namen eingeben, wonach das Programm wieder in zwei Teile geteilt wird.
   Nun die Ziel ("Destination-") Disk einlegen.
   Screens werden mit "name"+"0" und die beiden Programmblocks mit "name"+"1" bzw."name"+"2" abgesaved.
7. Nun sollte man einen Loader wie nachfolgenden schreiben:

```
10 CLEAR 27295
20 LET dos=15363
*30 RANDOMIZE USR dos:REM:LOAD "name0"CODE  *(wenn mit Screen)
40 RANDOMIZE USR dos:REM:LOAD "name1"CODE
50 CLS (optional)
60 RANDOMIZE USR dos:REM:LOAD "name2"CODE
70 REM hier kann man Pokes einfügen
80 RANDOMIZE USR 16384
```

8. Nun den Loader auf die Disk saven. Ich wünsche nun viel Spaß mit dem Programm.

WENN DAS GESPLITTETE PROGRAMM NICHT LÄUFT:

a. Manchmal benutzen die Programme den Interruptmode 2, (z.B. Tomah*wk) und man muß mit Option 5 den Interruptmode im RUNner MC -Program (Zeile 9100-9130) abändern. (So wie wenn man ein "$" in den Namen eines MAGIC Files einfügt).
b. Manche 16k Programme testen ob 48k frei sind und verteilen sich dann im Speicher, sodaß solche 16k Programme auch als 48k Programme abgesaved werden müssen.
c. Die Diskette ist voll, sodaß kein Platz für die gesplitteten Teile mehr ist.
d. Du hast den Magic Button im falschen Augenblick gedrückt, z.B. wenn das Programm testet was auf dem Screen ist oder so ähnlich. Hier hilft nur es an einer anderen Stelle nochmal zu versuchen !
e. Ja, und einige Programme (z.B. Nod*s of Y*sod,Cyber*n) wollen es auch nicht, wenn der Interrupt freigegeben ist, wenn sie ablaufen. In diesem Fall muß dann ähnlich wie in Punkt a beschrieben der Interrupt innerhalb des RUNner-Programms gesperrt werden. (DI)

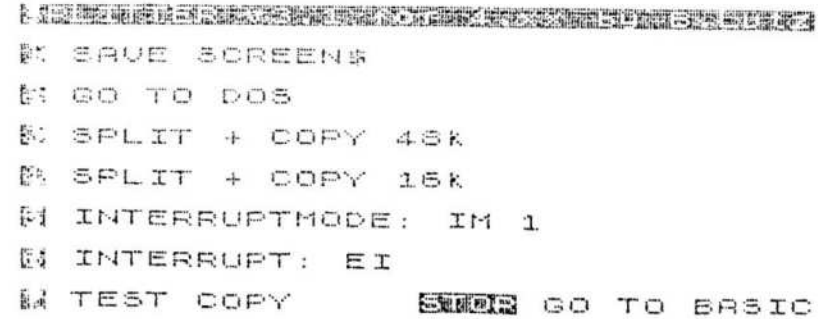
ÜBER DAS POKEN:

Die Adressen 27296-65535 können wie normal gepoked werden.
Die Adressen 23296-27295 sollten als POKE (xxxxx-6873),YY gepoked werden, wobei xxxxx für die normale Adresse steht und YY für den einzupokenden Wert !

P.S. das Programm wurde von einem Freund von mir auch für die Beta-Disk v5.xx umgeschrieben. Interesse ?

```
SPLITTER made by
  Bernhard Lutz, Hammerstr.35, 6729 Bellheim, Tel. 07272-71444
  10 CLEAR VAL "26999": LET dos=VAL "15363": LET im=VAL "86": LE
T ei=VAL "251"
 100 CLS : PRINT  INVERSE VAL "1";"SPLITTER v3.1 for 4.xx by B.L
utz"
 110 PRINT '"1  SAVE SCREEN$ "''"2 GO TO DOS"''"3 SPLIT + COPY 4
8k"''"4 SPLIT + COPY 16k"''"5 INTERRUPTMODE: IM ";"1" AND im=VAL
 "86";"2" AND im=VAL "94";''"6 INTERRUPT: ";"EI" AND ei=VAL "251
";"DI" AND ei=VAL "0"''"7 TEST COPY";TAB VAL "16";"STOP  GO TO B
ASIC"
 120 LET A$=INKEY$: IF a$=" STOP " THEN  RANDOMIZE USR VAL "0"
 130 IF a$<"1" OR a$>"7" THEN  GO TO VAL "120"
 140 GO TO VAL a$*VAL "100"+VAL "100"
 200 LET ix=VAL "32768": LET t=VAL "1": LET s=VAL "0": LET l=VAL
 "27": GO SUB VAL "1000"
 210 GO SUB VAL "4000": GO SUB VAL "5000": LET ix=VAL "16384": L
ET n$=v$+"0": LET l=VAL "6912": GO SUB VAL "1200"
 220 RUN
 300 RANDOMIZE USR VAL "15360"
 310 RUN
 400 LET mem=VAL "38240": GO TO VAL "510"
 500 LET mem=VAL "5472"
 510 GO SUB VAL "3000"
 540 GO SUB VAL "2100": GO SUB VAL "2000": GO SUB VAL "4000": LE
T ix=VAL "27296": LET l=mem: LET n$=v$+"1": GO SUB VAL "1200"
 550 LET ix=VAL "16384": LET l=VAL "4040": LET n$=v$+"2": GO SUB
 VAL "1200"
 560 RUN
 600 IF im=VAL "86" THEN  LET im=VAL "94": GO TO VAL "100"
 610 LET im=VAL "86": GO TO VAL "100"
 700 IF ei=VAL "251" THEN  LET ei=VAL "0": GO TO VAL "100"
 710 LET ei=VAL "251": GO TO VAL "100"
 800 CLS : LET l=VAL "22": PRINT "1 TEST 48k COPY "''"2 TEST 16k
 COPY "
 805 PRINT ''"INTERRUPTMODE: IM ";"1" AND im=VAL "86"; INVERSE V
AL "1";"2" AND im=VAL "94";''"INTERRUPT: ";"ENABLED" AND ei=VAL
"251"; INVERSE VAL "1";"DISABLED" AND ei=VAL "0"
 810 LET a$=INKEY$: IF a$<>"1" AND a$<>"2" THEN  GO TO VAL "810"
 820 IF a$="1" THEN  LET l=VAL "150"
 830 LET ix=VAL "27136": LET t=VAL "3": LET s=VAL "10": GO SUB V
AL "1000"
 840 GO SUB VAL "3000"
 850 GO SUB VAL "2000"
 860 GO SUB VAL "1600"
 870 RANDOMIZE USR VAL "16384"
1000 GO SUB VAL "1500": POKE VAL "27006",INT (ix/VAL "256"): POK
E VAL "27005",ix-VAL "256"*INT (ix/VAL "256"): POKE VAL "27008",
t: POKE VAL "27010",s: POKE VAL "27012",l: RANDOMIZE USR VAL "27
000": RETURN
1200 RANDOMIZE USR dos: REM : ERASE n$CODE
1210 RANDOMIZE USR dos: REM : SAVE n$CODE ix,l
1220 RETURN
1500 RESTORE VAL "9000": FOR a=VAL "27000" TO VAL "27043": READ
b: POKE a,b: NEXT a: RETURN
1600 RESTORE VAL "9100": FOR a=VAL "16384" TO VAL "16422": READ
```

```
b: POKE a,b: NEXT a: RETURN
2000 LET ix=VAL "16423": LET t=VAL "2": LET s=VAL "11": LET l=VA
L "16": GO SUB VAL "1000": GO SUB VAL "1600": RETURN
2100 LET ix=VAL "27136": LET t=VAL "3": LET s=VAL "10": LET l=VA
L "150": GO SUB VAL "1000": RETURN
3000 LET ix=VAL "22200": LET t=VAL "0": LET s=VAL "0": LET l=VAL
 "1": GO SUB VAL "1000"
3020 LET sp1=PEEK VAL "22209": LET sp2=PEEK VAL "22210"
3030 RETURN
4000 INPUT "ENTER NEW NAME: "; LINE v$
4010 IF LEN v$<VAL "1" OR LEN v$>VAL "7" THEN  GO TO VAL "4000"
4020 INPUT "INSERT DESTINATION-DISK          THEN PRESS ENTER: ";
 LINE e$
4030 RETURN
5000 RESTORE VAL "9200": FOR a=VAL "40000" TO VAL "40011": READ
b: POKE a,b: NEXT a: RANDOMIZE USR VAL "40000": RETURN
8999 STOP : REM LOADER 27000,44
9000 DATA VAL "205",VAL "6",VAL "60",VAL "229",VAL "33",VAL "0",
VAL "64",VAL "22",VAL "1",VAL "30"
9010 DATA VAL "0",VAL "62",VAL "27",VAL "6",VAL "1",VAL "14",VAL
 "5",VAL "245",VAL "197",VAL "213"
9020 DATA VAL "229",VAL "205",VAL "253",VAL "59",VAL "225",VAL "
36",VAL "209",VAL "193",VAL "28",VAL "62"
9030 DATA VAL "16",VAL "187",VAL "204",VAL "160",VAL "105",VAL "
241",VAL "61",VAL "200",VAL "24",VAL "229"
9040 DATA VAL "30",VAL "0",VAL "20",VAL "201"
9099 REM RUNNER X,39
9100 DATA VAL "243",VAL "237",im,VAL "33",VAL "39",VAL "64",VAL
"17",VAL "0",VAL "91",VAL "1"
9110 DATA VAL "160",VAL "15",VAL "237",VAL "176",VAL "49",sp1,sp
2,VAL "241",VAL "237",VAL "79"
9120 DATA VAL "241",VAL "237",VAL "71",VAL "241"
9120>DATA VAL "241",VAL "237",VAL "71",VAL "241",VAL "225",VAL "
209",VAL "193",VAL "217",VAL "8",VAL "253"
9130 DATA VAL "225",VAL "221",VAL "225",VAL "225",VAL "209",VAL
"193",VAL "241",ei,VAL "201"
9199 REM LDIR 32768-16384
9200 DATA VAL "33",VAL "0",VAL "128",VAL "17",VAL "0",VAL "64",V
AL "1",VAL "0",VAL "27",VAL "237",VAL "176",VAL "201"
```



Das Programm ist gegen Einsendung einer Diskette 3,5" bzw. 5,25" (bitte in stabiler Verpackung und mit Angabe des Formats) plus Rückporto in Briefmarken bei mir direkt zu erhalten.

Greetings to DMC, Lord Luxor, SMT Mannheim/Ludwigshafen etc.

Manfred Messerschmidt, Wildtaubenweg 64, 7050 Waiblingen

Datenmanager 87 / kleine Veränderungen mit nützlichen Effekten

Konfiguration: Spectrum 48K; Opus Discovery

Steckbrief: Der Datenmanager ist ein leistungstarkes Datenverarbeitungprogramm mit einem sehr großen Datenspeicher. Maximal können über das Indexverfahren 1500 Datenblätter verwaltet werden. In Blöcke von jeweils 5000 Bytes sind die Daten auf Diskette abgelegt. Je nach Datenmenge bzw. Größe eines Datenblattes, variiert innnerhalb eines Blockes, die Anzahl der Datenblätter.
Über dem Maskengenerator läßt sich indiviuell eine Bildschimmaske erstellen.
Neben den Dateien verarbeitet und verwaltet er auch 28 Programmteile Nach Aufruf der jeweiligen Option, legt sich das aktuelle Programmteil in das Grundprogramm ein. Der nicht mehr benötigte Teil wurde zuvor aus dem Progr. entfernt.
Diese raffinierte Vorgehensweise beherrscht das normale Betriebssystem nicht. BETA BASIC 1.8 mit einer zusätzlichen 64-Zeichen Routiene unterstützt das Sinclair Basic. Durch die im Programm benutzten Prozeduren ist eine Beschleunigung des Ablaufes gewährleistet.

Bezugsquelle: Kai Uffenkamp, Gartenstr.3, 4904 Enger/Dreyen

Programmiert von: Heinz Müller

Upate: Vom Januar 1989
Der Datenmanager 87 ist eine Weiterentwickung des Datenmanager 84. Bestimmt gibt es zwischen dem DM 84 und DM 87 einige Versionen. Dateien, welche vom DM 84 stammen, können über ein Unterprogramm konvertiert und mit dem DM 87 weiter benutzt werden.

Handhabung: Durch den praktischen Aufbau mit BETA BASIC lassen sich jederzeit Veränderungen vornehmen. Das Programm läßt sich breaken. Eingaben oder Änderungen werden vom Progr. nicht akzeptiert. Um diese Problematik zu umgehen, verwendet man für die Korrektur das normale BETA BASIC 1.8.

Wenden wir uns dem ersten Programmteil zu:

Name: MANAGER

Dieser Programmteil liegt bei den meisten Optionen im Rechner. Er überwacht und verwaltet die Eingabe- und Suchprogramme.

Orginal-Länge:	7784	nach dem Delete:	6625
Neue Prgr.-Länge:	7819	nach dem Delete:	6388
		Ersparniss:	237 Bytes

Mit dem Zusammenfassen der Zeilennummern läßt sich Speicherplatz einsparen. BETA BASIC besitzt hierfür den Befehl JOIN. JOIN verbindet die aufgerufene Zeilennummer mit der nachfolgender Zeile. Die integrierte Zeile verliert ihre Statusnummer. Sie ist mit einem

Doppelpunkt von der ersten getrennt.

Hier nun alle zusammengefaßte Zeilen:
85+86; 170+173; 180+190; 192+193; 801+802+805+810; 1760+
1770+1772; 1790+1810; 1820+1830; 1840+1850+1860; 2480+2490;
8200+8201+8202+8203; 8300+8310; 8320+8330; 8505+8510;
9101+9110

Folgende REM Zeilen sind aus dem Programm zu entfernen:
8500; 8600; 8700; 8800

Verändertes Listing
****

Kommentar
****

```
62 IF iy>22 THEN PROC z4
```

```
/ Der DM 84 arbeitet mit 22 Zeilen. Im DM 87 sind 21 Zeilen
/ vorgesehen. Mit einer kleinen Abänderung können weiterhin
/ 22 Zeilen benutzt werden.
```

```
96 LET y$=USING$("0000",LEN w$/2
 )+" Sätze gefunden": LET iy=VAL
  "22": LET ix=VAL "VAL "63"-LEN
  y$: PROC d1: END PROC
```

```
/ Zeilen zusammengefaßt und platzsparend verändert.
/ Die Zeilen-Nummer 97 ist mehr existent.
```

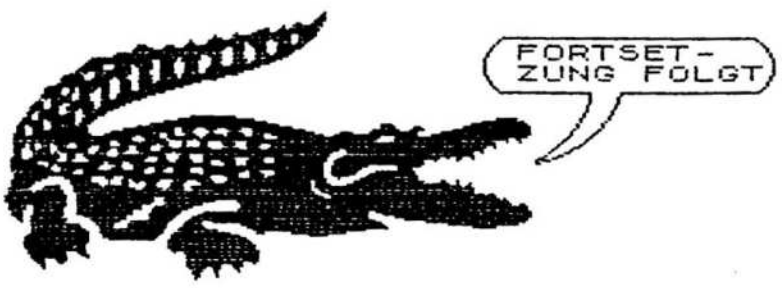
```
110 CLS : LET c=63: LET z=AT:LET
  y$="10000060063Katalog der Dis
 kette (J/N)": PROC prompt: PROC
 geth: IF H$="N" THEN PROC lo: G
 OTO 131
111 PROC lo: GO TO VAL "113"
112 LET h$=" M = Zurück zum Menu
 e andere Taste Wiederholung des
 CAT! ": LET EF=NOT PI GO SUB 11
 5: GOTO 125
113 PRINT AT NOT PI,NOT PI; "DAT
 EIEN EINLESEN ": LET EF=NOT PI:
115 PROC prt
120 POP
125 PROC geth: IF h$<>"M" THEN P
 ROC c: GO TO 113
126 PROC mes4: LET zr=126: LOAD
 *1;"MENUE"
130 PROC c: LET z=21: PROC c: PR
 OC cat
131 LET y$="21000060063Name der
 Einlesedatei :": PROC input: LE
 T f$=x$
135 LET zr=112: PROC load: DELET
 E 7000 TO 7066: DELETE 100 TO 1
 35
```

```
/ Dieser Block wurde fast vollständig neu geschrieben.
/ Normalerweise zeigt das Programm beim Einstieg oder beim
/ Umstieg in eine andere Datei immer den Katalog an.
/ Unverhältnissmäßig lange Zeit benötigt der Katalog für eine
/ Auflistung. Die Neu eingesetzte Auswahlroutine frägt jetzt
/ nach dem Katalog.
/ Bei einer Fehlermeldung erscheint ebenfalls ein neuer
/ Hinweis.
/ Die Zeile 112 ist mit einem schwarzen Balken und aufge-
/ hellter Schrift dargestellt. Folgendermaßend tippt man sie
/ ein: Nach dem Anführungszeichen,
/       INV. VIDEO: in den E-Modus gehen und eine 9 drücken
/
/ Jetzt kommt der Text.
/
/ Bevor das 2. Anführungszeichen gesetzt wird,
/       in den E-Modus gehen und eine 8 drücken; TRUE VIDEO
/
/ Bewußt besitzt dieser Block keine platzsparende Maßnahmen,
/ da dieses Teil im Normalbetrieb durch DELETE entfernt ist.
/
/
/
```

```
265 GO SUB VAL "299": PROC delet
e: PROC RESET: GO TO VAL "170"
```

```
/ Einsprungzeile wegen den Unterprogr. (Such- und Eingabe-
```

```
                                        / files) verändert. Zeilen-Nr. 270; 280 nicht mehr existent
                                        ============================================================

2010 LET Y$=2122001006 3Nummer de       ============================================================
 r Submaske :": PROC input: LET         / Zeilen zusammengefaßt, den Löschbefehl für INFO-Zeile her-
 sub=VAL x$: LET zr=VAL "2010":         / ausgenommen. Anstelle des eigständigen Löschbefehls,
 PROC load: LET z=VAL "21": PROC        / greift man den allgemeinen zurück. Die Zeilen-Nummern
 c: END PROC                            / 2020; 2030; 2040 sind nicht mehr existent. Variable für die
                                        / Zeilenlöschung eingefügt.
                                        ============================================================

2455 LET zr=VAL "2455": PROC mes        ============================================================
 3: LET m$=USING$("0000",inbc):         /
 LOAD * drive;f$+m$ DATA i$():          / Platzsparend verändert
 LET ip=SGN PI: LET inbc=inbc+SG        /
 N PI: END PROG                         ============================================================

7066 DEF PROG lo: LET z=10: PROG        ============================================================
   c: END PROG                          / Zeile ist nur kurzzeitig aktiv; im Normalbetrieb ist sie
                                        / durch DELETE herausgenommen; Variable für die Zeilen-
                                        / löschung eingefügt.
                                        ============================================================

8100 DEF PROC mes1: LET h$="(W)         ============================================================
 eiter suchen oder (E)nde": LET         / Zeilen zusammengefaßt;  Zeilen-Nr. 8105 nicht mehr existe
 c=VAL "30": PROC prt                   ============================================================

8150 DEF PROC mes3: IF PEEK VAL         ============================================================
 "65522" AND NOT dat THEN LET h         / Zeilen zusammengefaßt;     Zeilen-Nummer 8155 nicht mehr
 $="Bitte Daten-Disk => Taste <W        /
 >": LET c=VAL "30": PROC PRT: P        / existent;     Variable für die Zeilenlöschung eingefügt
 ROC w: PROC geth: IF h$="W" THE        /
 N PROC C: LET DAT=SGN PI: ELSE         ============================================================
 PROC MES3

8165 DEF PROC mes4: IF PEEK VAL         ============================================================
 "65522" AND DAT THEN LET h$="Bi        / Zeilen zusammengefaßt;     Zeilen-Nummer 8170 nicht mehr
 tte Programm-Disk => Taste <W>         /
 ": LET c=VAL "30": PROC Prt: PR        / existent;  Variable für die Zeilenlöschung eingefügt
 OC w: PROC geth: IF h$="W" THEN        /
  PROC C: LET DAT=NOT PI: ELSE P        ============================================================
 ROC MES4
```



F.M.MOCZKO
Haselsteig 41
1000 Berlin 47

Berlin , 20.03.90

SCHREIBEN UND DRUCKEN (TEIL IV)

Schreiben und Drucken aus "VU-File"

Ein Programm zum Schreiben, Sortieren und Drucken habe ich Euch als zwar zuverlässiges, aber eben doch langsames Basic-Programm in Teil II vorgestellt (in Heft #73).

In Heft 10 (1986) , Seite 88 der Zeitschrift <Happy Computer> findet man unter dem Titel : " Tasword in Reih und Glied " einen Artikel über alphabetisch sortierte Tasword-Dateien, mit dem man alle Eingaben in alphabetischer Reihenfolge sortieren (und dann anschließend auch drucken) kann.
Viel effektiver ,schneller und sicherer aber ist das Sortieren und Drucken aus VU-File:
Dieses Programm dient dazu, Eure Adressen, Videofilme, Schallplatten , Zeitungsartikel zu sortieren und bei Bedarf schnell wiederzufinden.
"VU-File" erzeugt "records" , die eine universelle Bearbeitung erlauben und besitzt eine Suchroutine , die mit hoher Geschwindigkeit arbeitet.

Auch zum Erstellen eines Inhaltsverzeichnisses ist "VU-File" sehr gut geeignet.

Da ich via IF_1 den Drucker betreibe, mußte das Programm an die serielle Schnittstelle angepaßt werden:

a) "VU-File" normal von der Cassette laden, nach dem Laden des Basic - Teils das Programm mit "BREAK" unterbrechen, und das folgende Programm eingeben :

```
 50>FORMAT "b",9600:OPEN #3,"b":BORDER 1:PAPER 1:CLEAR 25087:LOAD ""CODE

 100 INK 7: DIM f$(32): LET a=29785: GO TO USR 29734

1000 CLS : PRINT "Save a file": GO SUB 6000: GO TO USR a

1005 SAVE g$CODE s,1: PRINT "Press V to VERIFY "'"or any other key to continue."

1010 IF INKEY$="" THEN  GO TO 1010

1020 IF INKEY$<>"V" AND INKEY$<>"v" THEN  GO TO 1200

1100 GO SUB 7000: VERIFY g$CODE

1200 GO TO USR a

2000 CLS : PRINT "Load a file": GO SUB 6000: GO SUB 7000: LOAD g$CODE s: GO TO U
SR a

3000 LPRINT f$( TO 32): GO TO USR a

6000 INPUT "Enter file name: "; LINE g$: LET f$( TO 10)=g$: LET s=CODE f$(11)+25
6*CODE f$(12): LET l=CODE f$(13)+256*CODE f$(14): RETURN

7000 PRINT '"If the program stops executing  and returns to BASIC with an    err
or code then enter:"''"GO TO USR a"''"to get back into VU-FILE.": RETURN
```

b) Mit Save "VU-File" Line 50 wird das modifizierte Programm dann auf Band abgespeichert; den MC - Teil laden wir anschließend , z.B. mit Copyplus.

Als Beispiel sollen uns die Spiele-Artikel des "Happy Computer" - Jahrgangs 1986, dienen, d.h.: wir möchten am Ende die Liste in etwa folgender Form dem Drucker entnehmen :

TITEL	*HAPPY*HEFT(/86)	SEITE
AIRWOLF	02	175
ANTIRIAD	12	090
BORED..	08	159
CASTLE..TER	05	163
CASTLE..TER	07	168

Dazu laden wir zunächst das adaptierte Programm und rufen dann den Menu - Punkt 6 (Erase current file) auf ; Press "E": to enter a file.
RECORD LAYOUT
(mit Caps Lock) ergibt sich folgendes Bild
(unterstreichen mit dem - Zeichen)

TITEL	*HAPPY*HEFT(/86)	SEITE

Mit STOP in die DATA FIELDS
Mit den Cursor-Tasten den Startpunkt jeder einzelnen Spalte ansteuern, dann ENTER , dann 7 bzw. 0 (für Paper bzw. Ink), am Ende STOP (to exit)

Enter the record
Jetzt die Informationen eingeben, das Springen von Spalte zu Spalte erfolgt mit ENTER.

Nach Eingabe aller Daten : STOP (ins Hauptmenu)

Da wir zunächst die Records nach der 1.Spalte (Titel des Artikels) ordnen und später drucken wollen, drücken wir O(rder) und das A von Airwolf beginnt zu blinken (als Zeichen dafür, daß nach der 1.Spalte geordnet wird). Bestätigung durch ENTER.

Nun überprüfen wir die Voreinstellung des (noch nicht eingeschalteten !) Druckers : bei mir muß CR (AutoFeed XT) unbedingt auf "on" stehen.

Spectrum User Club Wuppertal

Manfred Nolten, Hauptstraße 20,4006 Erkrath=2 Erkrath,04.05.90

Hallo Freunde !

Das neue Info (5) habe ich heute erhalten und bedanke mich bei allen, die mich lobenswert erwähnt haben. Zuviel der Ehre eigentlich, mir liegt genau wie allen anderen daran, daß sich möglichst viele der Mitglieder am INFO beteiligen. Wie immer der Beitrag auch aussehen mag, mit Sicherheit findet er das Interesse von mindestens einem User, und damit hat es sich dann auch schon gelohnt!
Schade daß ich kein BETA System habe, die Beiträge von Herrn Wlecke sind echt umfangreich und kompetent gemacht. Hat einer der User schonmal diese Beiträge in Angriff genommen und damit gearbeitet/in eigene Programme eingebaut/ etc. ?
Bisher konnte ich noch keine Reaktion auf die Arbeit von Herrn Wlecke sehen. Ich möchte keine Komplimente für Herrn Wlecke "fischen", aber es würde Ihn doch bestimmt interessieren, wie seine Ausarbeitungen ankommen.
Jetzt zu der Frage von *Herrn Harald Abel*: Mehr CAT Einträge bei OPUS
Hierzu erst einmal ein paar Grundlagen : Die OPUS formatiert von Hause aus 40 Spuren, 1-seitig; pro Spur 18 Sektoren je 256 Bytes. Das macht 40*18*256,also 184.320 Bytes / 1024 = 180 KB Kapazität. Nach einer Formatierung zeigt der CAT 178 KB frei, effektiv sind das 178,25 KB.
Die ersten 7 Sektoren auf Spur 0 sind der Verwaltung der Diskette vorbehalten:
Spur 00/Sektor 00 = hier steht ein kurzes MC Ladeprogramm mit den Daten über die Diskettenstruktur(40/80 Spur,Datendichte(128/256/512 Bytes, Seitenanzahl/..m
Spur 00/Sektor 01 - 07 : Diese 7 Sektoren sind von Hause aus den CAT Einträgen vorbehalten. Jeder Sektor fasst 16 Einträge a 16 Bytes; jeder Eintrag hat 6 Bytes vorweg mit den Daten auf welcher Stelle der Disk das Programm ist, Länge, usw., die letzten 10 Bytes sind der Programmname(Falls kürzer als 10 wird automatisch HEX 20 gesetzt, also SPACE).
Theoretisch sind also 7*16 Einträge möglich = 112 . Da aber der jeweils ERSTE und der LETZTE Eintrag der Name der Diske ist, sowie die Daten über Gesamtanzahl Sektoren etc, bleiben faktisch 110 Einträge zur freien Verfügung.
Aus den genannten Zahlen ergibt sich dann auch der freie Speicherplatz auf der Diskette :
40 Tracks * 18 Sektoren * 256 Bytes = 184.320 Bytes/1024 = 180 KB; 8 Sektoren belegt (Spur 0/Sekt. 0-7) * 256 Bytes= 2048 Bytes; 184.320 - 2048 = 182.272 / 1024 = 178 KB.

Das ist nicht ganz richtig, wie oben genannt sind 178,25 KB frei! Das liegt daran, daß die OPUS intern MINUS 1 rechnet, also den Sektor 00 (MC Programm/Diskdetails) nicht mit - zählt! Also dann : 7 Sektoren (nur CAT) *256 = 1792 ;184.320 minus 1792 sind dann 182.528 Bytes / 1024 = 178.25 KB frei.

Dieses MINUS 1 darf man dann auch bei nachstehendem Programm nicht vergessen, es berücksichtigt als DEFAULT 6 Sektoren, was in umgekehrter Reihenfolge wieder die richtigen 7 sind. Das Programm öffnet Kanal 0 für " CAT " (E-Mode/Symb.Shift+ 9, nicht CAT in Buchstaben eingeben!) und reserviert die per Neueintrag auf der Disk die gewünschten Sektoren für CAT-Einträge. Hier sollte man vorher ausrechnen, wieviel Einträge benötigt werden : 6(+1) = 110 Einträge / Normalfall
7(+1) = 126 Einträge
Pro Sektor also 16 Einträge mehr, die beiden DISK-NAME sind ja schon bei den Normal 110 freien berücksichtigt.

Aber Achtung, je mehr CAT Einträge reserviert werden, desto weniger der frei Speicherplatz! Umgekehrt aber auch die Möglichkeit Platz zu schaffen : Auf eine 178 KB Disk gehen im Schnitt 3 Spiele ; bei MF1 Backups evtl. 4. Also entweder 4*Basic Lader / 4* 3 Codeteile (...1/...2/...3)/ plus 2*der Diskname, das sind 18 Einträge! Es werden daher nur 2 Sektoren für CAT Einträge benötigt!. In dem Falle könnte man mit der Routine dann 5 Sektoren zu je 256 Bytes sparen, das sind immerhin 1280 Bytes, wen haben nicht auch schon mal ein paar Bytes zuwenig ein "no room on Disk" gebracht.?

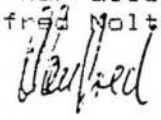
Also hier jetzt das Programm :

```
10 DEF FN a()=CODE INKEY$#0+256*CODE INKEY$#0
20 INPUT "Laufwerk Nr.:",d'"Anzahl Sektoren (6=Normal):",s
30 CLEAR #0:OPEN#0;" CAT ";d RND16:POINT#0;2
40 LET m=FN a()*0+FN a(): IF s>m THEN CLEAR #0:OPEN#0;d;"x" OUT 2^31:STOP
50 POINT #0;1:RANDOMIZE FN a()*FN a()*0+s:PRINT #0;CHR$ PEEK 23670;CHR$ PEEK
   23671;:CLOSE #0
```

Das ganze SAVE *1;"MEHR CAT" LINE 10. Zum Test leere Disk formatieren und CAT machen, 178 KB frei. Dann das Programm MEHR CAT laden, die formatierte Disk wieder rein und ausprobieren. Laufwerk angeben, Sektoren-Anzahl (6 Normal) und dann ist in Blitzesschnelle die Disk geändert! Viel Spass damit!

Zu der Frage von *Bernhard Lutz* Absturz/TASWORD :
Die Problematik hatte ich auch und habe eigentlich nur eine vermutete Lösung dazu : Entweder spinnt der Z80, die ULA, oder aber die RAMs! Ich hatte bei extensivem Gebrauch der Textverarbeitung (TW2 / TW3 / LAST WORD), und immer nur bei Textverarbeitung, die gleichen Probleme. Wechseln des Z80 und der ULA brachten keine Lösung, kann man aber im nachhinein nicht zweifelsfrei sagen. Die RAMs, bzw. der Refresh-Zyklus derselben, lässt mit dem Alter der Chips nach. So kann es glaube ich passieren, daß das scannen der Tastatur auf Eingabe (alle 1/50 Sekunde) und die dauernde Eingabe als solches mit den nicht korrekten Refresh Signalen durcheinander kommt und in Hängen bzw. Absturz des Rechners resultiert. Ich habe die RAM's gewechselt und seitdem keine Probleme mehr mit TW3. Mit Last Word und TW2 arbeite ich seit langem nicht mehr. Bei Wechsel der RAMs unbedingt die gleichen Typen wieder einsetzen, siehe auch Beitrag von Herrn Claus-Jörg Weiske in INFO 5! (OKI / TI). Ich hoffe Sie haben Erfolg mit den RAMs.

Das war alles für diesmal, herzliche Grüße,
Manfred Molten.



ZU VERKAUFEN:

1 Netzteil DM 20,00 inkl. Porto
1 Kassettenrekorder DM 20,00 inkl. Porto
Beta Basic 3.0 DM 30,00 mit dt. Handbuch inkl. Porto
Beta Basic 4.0 für den 128+ mit engl. Handb. DM 45,00 inkl. Porto
Tasword plus/Two für den 128er DM 30,00

Hermann Mayr, Grafstr. 2/4, 8025 Unterhaching

Text-o-Mat (5)

Herbert Hartig
Postfach 147,8938 Buchloe

Unser Textautomat hat jetzt (mit Kalenderroutine) noch 1800 freie Speicherplätze, 1100 davon können wir noch verwenden. Nehmen wir Rechenroutinen mit herein, dann kommen wir auf etwa 900 freie bytes für's Basicprogramm. 700bytes braucht aber das Microdrive für sich, mindestens 500 bytes die "Opus" Diskettenstation. Wir müssen also darangehen ein zweites Basicprogramm einzurichten, das gegen das erste ausgetauscht werden kann und die Routinen enthält, die wir im ersten nicht mehr unterbringen. Wichtige Routinen werden wir in beiden Programmen vorsehen,um nicht ständig umladen zu müssen.
Hat man mehr als zwei, so kann man in der Zeile, in der eine bestimmte Routine laufen soll, die aber in dieser Version nicht vorhanden ist, die betreffende Version aufrufen: LET b$= "run2": GO TO 2. Dann wird die richtige Version geladen und die Routine steht dann beim nächsten Aufruf zur Verfügung.
Man kann auch einzelne Routinen mit Merge nachladen, braucht aber dazu mehr freien Speicherplatz als beim Laden eines ganzen Programmes. Außerdem muß man dann auch immer wieder irgend etwas löschen, sodaß die Sache nicht besonders einfach wird.
Man kann auch auf verschiedene Disketten verschiedene Programme jeweils mir "run" speichern, das bestehende Programm jeweils mit NEW löschen und mit RUN das gewünschte Programm laden. Damit trotzdem der Text nicht verloren geht,fügen wir in Zeile 15 ein:
IF PEEK VAL "54784"= SGN PI THEN
Dann wird der Maschinencode nur geladen,wenn noch keiner im Speicher ist, außerdem wird das Textfile nicht gelöscht.
Wir fügen eine Zeile 18 ein, die unseren Drucker initialisiert.
Weitere Beiträge über Text-o-Mat werde ich in zwangloser Folge schreiben. Jeder kann dann das einbauen, was er brauchen kann und das weglassen, was er niht braucht.

Beschreibung der Rechenroutinen:
Diese stehen bei 4000-4054 und sind unkonventionell programmiert und wohl auch unmathematisch, aber gut brauchbar. Zahlen bis 99999,99 werden addiert und subtrahiert. Zahlen ab 100 000 werden in der ersten Spalte durch Buchstaben in (ASCII-Code -48) angezeigt. Nach 9 kommt also ":" als 10,";"als 11,"<"als 12 usw. Mit diesen Buchstaben rechnet der Computer auch noch richtig. Das spart Speicherplatz und Platz bei einer Zahlenkolonne. Die Endsumme kann ja dann korrigiert werden. g1234,33 ist also (103-48)› 551234,33. Wird die Zahl‹ 0, dann wird diese umgekehrt notiert, also -3 = /9997,00, vorne ein / › (48-47= -1). Auch mit dieser Zahl kann weitergerechnet werden.

EXT-r-"+" (Plus), Zeile 3012: Der Rechner fragt zu Beginn, ob er eine Gesamtsumme berechnen soll. Wird mit "j" geantwortet, dann schreibt er an eine vorher mit EXT-r-"0" bezeichnete Stelle die Summe aller folgenden Eingaben. Das gilt solange wir in diesem Modus bleiben und nicht zwischendurch ins Hauptmenue gehen.
Nach Eingabe "Betrag"schreibt er diesen in der Form 00100,00 für 100 an die Stelle, die der Cursor einnimmt u.zw.d.Komma. ↑
Gibt man an gleicher Stelle einen neuen Betrag ein, so wird dieser zum vorhandenen addiert. Gibt man als Betrag ENTER, so wird der gleiche Betrag nochmal addiert. Subtrahiert wird mit "-" vor dem Betrag.
Besonderheit:(Wird mit EXT-r-0 eine Stelle markiert und auf dieser mittels EXT-r-"+"gerechnet, dann wird der Betrag jeweils doppelt eingetragen.)
So kann man z.B. Zahlungslisten anlegen und in jeder Zeile jeweils eingehende Zahlungen addieren. z.B.

```
                                                  Meier     00100^00
                                                  Müller    00100,00
EXT-r-"=" (ist gleich), Zeile 3015: addiert       Schmidt   00050,00
Kolonnen. Dazu wird der Cursor auf das ers-       Schulze   00050,00
te Komma gesetzt (^),unten kommt eine Reihe >     ----------------
Der Rechner setzt die Summe darunter                        00300,00
(Wichtig ist nur das "-" unter der ersten Zahl oder Null.
Wird der Cursor auf eine tieferstehende Zahl gesetzt, so  werden
die höher liegenden nicht mitgezählt. Ist die  erste  Null  oder
Zahl nicht vorhanden, wird die ganze Zahl nicht mitgerechnet.

EXT-r-"^"(hoch), Zeile 3017: Ein durch EXT-r-0 markierter Betrag
wird an die Stelle des Cursors übertragen oder zu dem dort  ste-
henden Betrag addiert:  00100,00  markiert >  abgelegt>  00100,00
                        hier zu  100 addiert      >       00200,00
Besonderheit:(Wird auf die durch EXT-r-0 markierte Stelle  über-
tragen, dann verdoppelt sich jeweils der Wert,(Multipl. mit 2).
Mit diesen Routinen sind auch andere Berechnungen möglich,  z.B.
Multiplikationen, Kurswertberechnungen, Erträge/Verluste usw.
Ein Abdruck meiner bescheidenen Haushaltsführung mit Hund und  2
Katzen,anbei. Man kann so einen Abdruck seiner täglichen Kassen-
lage in der Geldbörse mit sich führen.

Ausgaben: 11.04.90                          Geld:soll  00077,71
                                          in der Börse ist   00075,35
Mo. Essen    Wäsche  Haushalt   Hund     Katzen  Sonstige   Gesamt
-------------------------------------------------------------------
J 00155,10 00029,00 00024,79 00025,88 00054,09 00079,55 00368,41
F 00163,85          00008,44 00021,35 00077,15 00139,87 00410,66
M 00295,57 00006,00 00004,47 00020,93 00068,80 00039,99 00435,76
A 00071,56 00035,60                   00025,80 00008,40 00141,36
  -----------------------------------------------------------------
S.A.686,08 00070,60 00037,70 00068,16 00225,84 00267,81 01356,19
```

Listing:

```
3001 PRINT BRIGHT SGN PI;INKEY$;
3005 IF INKEY$="0" THEN POKE VAL "32e3"+z2,VAL "44": LET z2=z1: POKE VAL "32e3"+z1,VAL "94": GO
TO VAL "3112"
3010 IF INKEY$="&" THEN GO TO VAL "7200"
3033 IF INKEY$="+" THEN GO SUB VAL "4e3": GO TO VAL "3112"
3035 IF INKEY$="=" THEN GO SUB VAL "4050": GO TO VAL "3112"
3037 IF INKEY$="^" THEN LET n=z1: LET z1=z2: GO SUB VAL "4003": LET m=Y: LET z1=n: GO SUB VAL "4
003": LET X=m: GO SUB VAL "4010": GO TO VAL "3112"
4000 GO SUB VAL "4002": INPUT "Betrag:";x$: GO SUB VAL "4008": LET z1=z1-PEEK VAL "62209"+VAL "8
9": GO SUB VAL "4002": GO SUB VAL "4010": LET z1=VAL "61": GO SUB VAL "4002": LET w=Y-X+VAL ".00
5": GO SUB VAL "4015": RETURN
4001 GO SUB VAL "4010": RETURN
4002 REM INPUT "Summej-saldo-mst.";d$
4003 LET z1=z1-VAL "28"
4005 IF PEEK (VAL "32023"+z1)=VAL "32" THEN LET Y=NOT PI: RETURN
4007 LET Y=((PEEK (VAL "32023"+z1))-VAL "48")*VAL "1e4"+((PEEK (VAL "32024"+z1))-VAL "48")*VAL "
1e3"+((PEEK (VAL "32025"+z1))-VAL "48")*VAL "100"+((PEEK (VAL "32026"+z1))-VAL "48")*VAL "10"+((
PEEK (VAL "32027"+z1))-VAL "48")+((PEEK (VAL "32029"+z1))-VAL "48")/VAL "10"+((PEEK (VAL "32030"
+z1))-VAL "48")/VAL "100": RETURN
4008 IF x$="" THEN LET X=m: GO TO VAL "4010"
4009 LET X=VAL x$: LET m=X
4010 REM IF d$="m" THEN LET w=Y*VAL "1.14"+VAL ".005": GO TO VAL "4015"
4012 LET w=X+Y+VAL ".005"
```

```
4015 LET w1=INT (w/VAL "1e4"): POKE (VAL "32023"+z1),w1+VAL "48": LET w=w-w1*VAL "1e4": LET w1=I
NT (w/VAL "1e3"): POKE (VAL "32024"+z1),w1+VAL "48": LET w=w-w1*VAL "1e3": LET w1=INT (w/VAL "10
0"): POKE (VAL "32025"+z1),w1+VAL "48": LET w=w-w1*VAL "100": LET w1=INT (w/VAL "10"): POKE (VAL
 "32026"+z1),w1+VAL "48": LET w=w-w1*VAL "10": LET w1=INT w: POKE (VAL "32027"+z1),w1+VAL "48"
4016 POKE (VAL "32028"+z1),VAL "44"
4020 LET w=w-w1: LET w1=INT (w*VAL "10"): POKE (VAL "32029"+z1),w1+VAL "48": LET w=w-w1/VAL "10"
: LET w1=INT (w*VAL "100"): POKE (VAL "32030"+z1),w1+VAL "48": RETURN
4050 GO SUB VAL "4002": LET Y1=Y
4052 LET z1=z1+VAL "64": IF PEEK (VAL "32023"+z1)=VAL "45" THEN LET z1=z1+VAL "64": LET w=Y1+VAL
 ".005": GO SUB VAL "4015": RETURN
4054 GO SUB VAL "4005": LET Y1=Y1+Y: GO TO VAL "4052"
```

Hubert Kracher
Kirchdorfer Str. 5
8201 Großholzhausen

****
* DOUBLE 'H' SOFTWARE *
****

Harald Lack
Heidenauer Str. 5
8201 Raubling

Hallo Rolf!!

Da wir der Ansicht sind, daß Computerei nicht nur eine todernste Sache sein soll, möchten wir Dir von unserer Seite folgenden Vorschlag machen:

In Anlehnung an "Die letzte Seite" der Wochenendausgabe der Süddeutschen Zeitung, kam uns der Gedanke, ob es nicht möglich wäre auch dem Clubinfo eine Seite mit ironischen Kurzgeschichten und Gedichten rund um den Computer hin und wieder hinzuzufügen.

Teile uns doch bitte Deine Meinung dazu mit oder stelle unseren Vorschlag im Info zur Diskussion.

Als Beispiel für einen Beitrag zur letzten Seite möchten wir Dir "Die Gesetze der Programmierung" vorstellen:

1. Jedes Programm, das läuft, ist veraltet.
2. Jedes Programm kostet mehr und dauert länger.
3. Ist ein Programm nutzlich, wird es geändert.
4. Ist es nutzlos, wird es dokumentiert.
5. Jedes Programm dehnt sich aus, bis es allen verfügbaren SpeicherPlatz füllt.
6. Der Wert eines Programmes ist proportional dem Gewicht seines Outputs.
7. Die Komplexität eines Programmes wächst so lange, bis sie die Fähigkeit des Programmierers übersteigt, der das Programm pflegen muß.
8. Erlaube es Programmierern, Programme auf Deutsch zu schreiben, und du erkennst, daß Programmierer nicht auf Deutsch schreiben können.

Natürlich sollen solche Beiträge nicht in jedem Info erscheinen und dann nicht mehr als 1 Seite des kostbaren Platzes beanspruchen.

Mit besten Usergrüßen.....

Spectrum&
Text-o-Mat (6)

Als Datei für wissensch.Arbeiten, Sachregister, Buchzitate Ideen Gedankenschublade, Zettelkasten usw:
Die mir bekannten, für den Spectrum verfügbaren Dateien erlauben nur jeweils kurze Einträge je Datensatz oder Eintrag (Item), z.B Masterfile mit 128 Zeichen, VU- File nur 32 usw.
Wenn man aber unter einem Stichwort einen größeren Text ablegen will, bietet sich folgende Möglichkeit an:
Wir speichern einen Datensatz mit bis zu ca. 250 Zeilen (16000) Zeichen unter einem Suchwort (Begriff) ab.
Wenn wir unter einem Begriff mehr Daten gesammelt haben, speichern wir diese auf einer zweiten Diskette unter dem gleichen Suchwort ab, usw.,nummerieren diese z.B. von 1-10.
Kommen mehr Suchworte zusammen, als auf eine Diskette gehen, legen wir eine Diskette A oder weiter B-z an und bekommen so eine 2- dimensionale Diskettenordnung (z.B. A1 - Z10).
Durch eine Vordatei (Liste) können wir leicht feststellen, wo sich das gewünschte Suchwort befindet.
Dasselbe machen wir ja auch mit Karteikarten und Aktenordnern.
Wenn wir beim Spectrum auch nicht mit Megbytes protzen können, so können wir uns auf diese Weise doch eine beachtliche Datei aufbauen. Die Daten eingeben muß ja jeder, da helfen auch keine Festplatten.
Ein geladenes File können wir nun ansehen, durchsuchen, Teile davon abdrucken, verändern usw., kurz, viel mehr damit machen, als mit einem der üblichen Dateiprogramme.
Wir haben aber auch in der TASLIST- Routine von Weinzierl ein komfortables Suchprogramm an der Hand und können damit nach verschiedenen Kriterien gezielt suchen, gesuchte Teile abspeichern usw:
Bei Aufruf Bright- modus "P" › Zeile 8064 wird die Suchroutine geladen und gestartet.(Wenn Brigth-modus nicht geht siehe Text-o -Mat (3)).
Vorher müssen wir in Zeile 1 (Editor)das Suchwort mit nachfolgendem & Schreiben:
Wir arbeiten wieder LIVE: in die erste Zeile › Spectrum$
CAPS+ENTER- P: Es meldet sich das Menue:

```
      TASLIST
 Johann Weinzierl Jun.
(1)Strich              setzt Strich unter jede gefund.Zeile
(2)REF.VOR             listet bis zu einem folgenden #
(3)REF.Rueck            "    "   "  "   vorhergehenden #
(4)ANGABEN    *VOR:0   Zeilenno.wird angegeben   *Zeilen vor
(5)GR/KL      *RUE:0   Groß/Kleinschreibung egal  *  "   rück
```

wird nun 3mal ENTER eingegeben springt der CURSOR auf das erste gefunden Suchwort.

Wir drücken die Taste 4 und ENTER. Eine 7 erhöht dann "VOR", die Zeilenzahl vor dem Suchwort, die angezeigt werden soll, um eins, eine 6 vermindert um eins.Nach ENTER wird auf gleiche Weise"RUE" die Zeilenzahl rückwärts vom Suchwort, erhöht. ENTER
wir erhalten nun als Ergebnis:

Die mir bekannten, für den Spectrum verfügbaren Dateien erlauben
Zeile 005 Spalte 028 Seite 001
Wenn wir beim Spectrum auch nicht mit Megbytes protzen können,
Zeile 021 Spalte 015 Seite 001
Wir arbeiten wieder LIVE: in die erste Zeile › Spectrum$

Zeile 035 Spalte 048 Seite 002

dieses wird mit Taste "t" hier (am Ende des Textes) angefügt.
Deswegen dürfen die einzelnen Files auch nicht voll sein (320Zei
len). Mit "e" kommen wir wieder in den Editor.
"s" speichert die ausgewählten Daten ab. "w" sucht weiter, falls
noch nicht alles gefunden.
Ein weiterer Aufruf der Suchroutine erfolgt jetzt mit Bright- 0
(null), da diese nicht mehr geladen werden muß.

Beispiel für (1)+(3mal ENTER)

Die mir bekannten, für den Spectrum verfügbaren Dateien erlauben

---

Wenn wir beim Spectrum auch nicht mit Megbytes protzen können,

---

Wir arbeiten wieder LIVE: in die erste Zeile › Spectrum$

Wir können hier nun auch die erweiterten Funktionen von EXT- r
‹= siehe Text-o-Mat (2) › Zeile 3027, hier erweitert
‹› " ‹ " 3029, "
auf beliebige Position,Zeilenzahl und SAVE verwenden.

Mit Bright-modus "SPACE",Zeile 8096 ist auf Zeile 300 im Editor
ein Zettelkasten erreichbar, mit dem während einer anderen Ar-
beit getrennt Gedanken, Ideen usw. festgehalten und mittels EXT-
r- › abgespeichert werden können.

Listing:

```
3027 IF INKEY$="›" THEN GO SUB VAL "7010": LET b=VAL "32e3"+z1: GO SUB VAL "7015": PRINT USR VAL
 "55308",b,VAL "56128"+VAL "64"*ze,VAL "64"*zu: GO SUB VAL "7020": GO TO VAL "3112"
3029 IF INKEY$="‹" THEN GO SUB VAL "7010": LET b=VAL "56128"+VAL "64"*ze: GO SUB VAL "7015": PRI
NT USR VAL "55308",b,VAL "32e3"+z1,VAL "64"*zu: GO TO VAL "3112"
7010 INPUT "Zeile?w";z$: IF z$="" THEN LET z$="-1"
7011 IF z$="w" THEN LET z$=STR$ (ze+zu)
7012 INPUT "Zahl?";y$;"adr. SAVE ";x$;: IF y$="" THEN LET y$="1"
7014 LET ze=VAL z$: LET zu=VAL y$: RETURN
7015 IF x$="s" THEN INPUT "Name:";n$: LET r=SQR PI: SAVE *r;n$CODE b,VAL "64"*zu: GO TO VAL "311
2"
7016 RETURN
8000 RANDOMIZE USR VAL "54868": RUN
8064 LOAD *"m",SGN PI,"Taslist"CODE : GO TO VAL "8e3"
8096 LET a=USR VAL "64333": GO TO EXP PI
8200 INPUT "Name:";n$: SAVE *SGN PI;n$CODE VAL "53248",VAL "1408": RANDOMIZE USR VAL "54937"
```

Herbert Hartig
Postfach 147,8938 Buchloe

Spectrum User Club Wuppertal

RUPERT HOFFMANN
Steinwaldstr. 14
8591 FRIEDENFELS
Telefon: 09683/763

Hallo, liebe Specci-Freunde

endlich finde ich mal wieder Zeit, ein paar Worte an Euch zu richten. Ich wollte schon laengst schreiben, aber mir fehlte irgendwie die Zeit. In der Zwischenzeit hatte ich Urlaub und wenig spaeter mit zwei Frauen, naemlich Angina und die Influenza, zu tun. Aber nun sind diese Krankheiten ueberstanden. So aber Schluss damit, rein in Specci-Angelegenheiten!

Manfred Messerschmitt und Rolf Knorre bemerkten, dass es in der DDR ~~es~~ wahrscheinlich etliche Specci-User gibt. Ich habe seit Dezember mit einem Specci-User aus (Ost)-Berlin brieflichen Kontakt ~~mit ihm~~. Z. Zt. hat er Pruefungen zum Elektronik-Assistenten abzulegen. Zu meiner Adresse kam er dadurch, dass er auf Umwegen zu einigen Infos unsers User Club Wuppertal kam. In der Ausgabe, in der ich nach einem Schaltplan fuer den GP 50-S suchte, fragte er nach, ob ich evtl. schon einen hatte, da auch er einen gleichnamigen defekten Drucker hat. Ich schickte ihm Fotokopien vom Schaltplan; inzwischen funktioniert sein Druder wieder.
Nun hoffe ich bzw. er=Guido Abel um Hilfe!!! Er sucht ein CAD-Platienlayoutprogramm mit dem Namen

L A Y O U T 8 7 .

Wer kann ihm bzw. mir helfen. Da ich guten Kontakt zu ihm unterhalte, koennt Ihr euch auch an mich wenden!!!

So, in einem Info fiel das Wort Scanner. Einige fragten mich schon danach. Wer es noch nicht wissen sollte, was dieses Geraet ist, und in welchen "Variationen" es gibt, so hilft der folgende Bericht, falls ihn Rolf veroeffentlichen sollte, zu verstehen, was ein Scanner ist. Achtung, Rolf, beigefuegte Ablichtung hat auch auf der Rueckseite einen Text.

Der Aufruf von Manfred Nolten hat und sollte noch weitere positive Auswirkungen haben. Mit der Fragebogenaktion (wer arbeitet ihn aus?) sollte man dafuer sein. In einem solchen Fragebogen wird ja schliesslich nicht danach gefragt, ob man schon alle Kinderkrankheiten oder Mhiskuoperationen hinter sich hat.

Ich habe festgestellt, dass einige Druckroutinen fuer die Beschriftung von Video-Labels und Adressetiketten gebracht werden. Na, da will ich auch meinen Senf dazu beitragen. Ich benutze Endlos-Adressetiketten der Firma Zweckform im Format 88,9 x 35,7 mm. Als Drucker wird der SEIKOSHA GP-700 A verwendet. Ein Wort sollte nicht laenger als 16 Buchstaben sein, sonst drueckt er ueber den Rand hinaus. Listing am Ende dieses Textes.

Demnaechst werde ich mir einen EPSON-kompatiblen Drucker zulegen, um das Text-O-Mat-Programm von Hebert Hartig, das wirklich mit wesentlich teueren Programmen von GRosscomputern mehr als wetthalten kann, voll nutzen zu koennen.
Warm hab' ich mir 1986 den GP 700 A gekauft? Ganz einfach, es ist ein Farbdrucker; ich mache ja Glueckwunschkarten in Zusammenarbeit mit Art Studio. Um Graphik auf diesen Seikosha zu bringen, verwende ich ein spezielles Druckertreiberprogramm.

So, ich hoffe, dass dieser Beitrag bald im Info erscheint, aber die Post streikt leider, z. T. auch in meiner Region!!!
Das naechste Mal versuch' ich mal, ein paar Listing von Kurzprogrammen fuer unseren Specci zu bringen.

Tschuess, bis zum naechsten Mal!
Ein treuer Specci-User

Rupert

```
  10 REM ***Erstellung von            ADRESSETIKETTEN       der Groesse 88,9 x 3
,7 mm***
  20 REM ***Druckroutine***
  30 REM Einstellung der Breitschrift
  40 CLEAR #
  50 OPEN #3;"t": OPEN #4;"b"
  60 PRINT #4;CHR$ 27;"E"
  70 PRINT #4;CHR$ 14
  80 LPRINT : REM Anrede
  90 LPRINT : REM Name,Vorname
 100 LPRINT : REM Strasse, Post-fach
 110 LPRINT : REM Leerzeile
 120 LPRINT : REM PLZ, Ortschaft
 130 LPRINT : LPRINT : LPRINT : LPRINT : REM Leerzeilen bis zum naechsten Etik
t
 140 GO TO 80


  10 REM ***Druckroutine***
  20 REM Einstellung der Breitschrift
  30 CLEAR #
  40 OPEN #3;"t": OPEN #4;"b"
  50 PRINT #4;CHR$ 27;"E"
  60 PRINT #4,CHR$ 14: LPRINT . REM Text
  70 STOP
  80 REM Zurueckstellung in die Normalschrift
  90 PRINT #4;CHR$ 27;"F"
```

Fischeln, den 15.05.90

Ludwig Halbritter
Hoebuergen 30, 4
4055 Fischeln
Tel. 02151/4045

Hallo Spectrum-User,
nach ca. zweimonatiger Abstinenz möchte ich zum ersten Mal etwas für das Info beisteuern.
Wir hat sich nicht schon geärgert, dass viele Spiele nur wenig taktisch zu spielen sind. Oder eine kleine Geschicklichkeit im einen Joystick an die Tastatur anzuschließen. Die Spectrum-tastatur besteht aus acht Halbreihen zu je fünf Tasten, wobei die Tastenbelegung je Halbreihe von innen nach aussen erfolgt. Beim Drücken einer Taste wird der Stromkreis zwischen Halbreihe und entsprechender Spalte der Tastenmatrix geschlossen. Der Leiter nun vorsichtig ein mehrpoliges Computerkabel an den 8-poligen und den 5-poligen Tastaturanschluß auf der Rückseite der Platine. Das Kabel führen wir zu einer 13-poligen Buchse, die wir an die Gehäuse des Spectrums einfassen. Über einen passenden Stecker wird das Kabel an eine Peripherieeinheit gelötet. In die Buchse werden je 5 Kontaktsteckerbuchsen über jede der acht Halbreihen und der fünf Spalten eingelötet. Die Kabel für die fünf Spalten werden zweimal unter einen Widerstand von 470 Ohm angeschlossen. Nun wird noch ein Joystick folgendermaßen angebaut. Von jedem Kontaktstift werden die beiden Anschlüsse eines einfachen Gehäusesteckers gelötet und mit einem farblich gekennzeichneten Kabel nach außen geführt. Am besten verwendet man für jede Funktion eine andere Farbe. Sehr gut fürt sich das Kabel eignet sich der Joystick "Perfect Pro" von der Firma Konrad Electronic. Über die Gehäusestecker kann man jede Funktion des Joysticks mit der gewünschte Taste belegt werden.

Nun noch ein Hilfetip. Ich benutze ein deutsches Handbuch über den Spectrum 128K und eine Programmsammlung von Data Becker (Kopie vom Original, Umkosten werden selbstverständlich erstattet).

Mit freundlichen Grüssen

L. Halbritter

So funktioniert ein Scanner

Moderne Techniken im Druckgewerbe

Von Thomas Lünendonk

Der nachfolgende Artikel ist in der Zeitschrift „bit" erschienen und wird mit freundlicher Genehmigung abgedruckt.

To scan: absuchen, ablasten. Der Langenscheidt gibt Aufschluß über die Funktion eines zeitgemäßen Peripherie-Gerätes für Computer. Es geht um Scanner, die bei der elektronischen Bildverarbeitung den Input grafischer Informationen in den Rechenvorgang bewirken. Die Geräte „lesen" Bildvorlagen und setzen sie in digitale Datenströme um, die vom Computer verstanden und bearbeitet werden können. Die Bilder können im Rechner gespeichert werden und lassen sich zumeist in andere Programme, beispielsweise für Desktop-Publishing oder Desktop Presentation, übernehmen.

Auf dem Scanner-Markt existieren drei technische Typen: **Hand-, Einzug- und Flachbett-Modelle.** Handscanner werden, worauf ihr Name schon hinweist, über die zu scannende Vorlage per Hand gezogen. Einzug-Scanner ziehen eine Vorlage ein und lesen sie mit einer starren Abtastvorrichtung. Bei der Flachbett-Technik, wie sie Agfa einsetzt, werden die Originale auf die Glasvorlage des Scanners gelegt und von einem beweglichen Schlitten abgetastet. Dieses Verfahren hat zwei wesentliche Vorteile: Für die Vorlage besteht keine Gefahr der Beschädigung, außerdem können auch Bücher, Atlanten oder andere mehrseitig und großformatige Vorlagen bearbeitet werden.

Scanner unterscheiden sich jedoch auch in ihrer Auflösung und Grauwertwiedergabe. An der Spitze der höchstauflösenden Geräte rangiert z.B. der Agfa-Scanner Focus II. Seine Auflösung beträgt bis zu 800 Dots per Inch (dpi). Weiterhin verfügt er über eine Abbildungstiefe von 256 Graustufen pro Bildpunkt (Pixel).

Im Gegensatz zu den Grafiken, die zumeist mit Business-Grafik- oder Mal-Programmen am Bildschirm entstehen, lassen sich mit Hilfe des Scanners auch „fertige" Illustrationen – Analogvorlagen – per Rechner verwerten. Hier fungiert der Scanner als das **„Auge" des Computersystems,** mit dem Firmenlogos und Wappen, Landkarten-Ausschnitte oder auch Fotografien in ein Programm eingelesen werden.

Beim Planen eines Layouts lassen sich Originalfotos so für das Scribble manipulieren, daß der Betrachter einen realistischen Eindruck vom Endprodukt erhält. Einige Scanner, beispielsweise die Agfa-Focus-Scanner, verfügen über die Wahlmöglichkeit der Auflösung. Sie reicht von 75 bis 800 dpi. Das heißt, die jeweilige Vorlage wird mit maximal 800 Punkten pro Zoll von einer elektronischen Leuchtdiodenzeile abgetastet. Die niedrigen Auflösungen, die weniger Bearbeitungszeit erfordern, empfehlen sich in jenen Fällen, in denen eine Vorlage nur schemenhaft in einem Groblayout erkannt werden muß, die hohen Auflösungen für hochwertige Layouts oder Endprodukte.

Software wichtig

Mit Grafikprogrammen lassen sich unterschiedliche Vergrößerungsmaßstäbe der Vorlage am Bildschirm realisieren und manipulieren. Auch in den Scan-Software-Programmen McView Plus und PCView Plus von Agfa ist schon eine Vielfalt von Gestaltungs- und Manipulationswerkzeugen enthalten. Dazu gehören unter anderem die Justierung von Kontrast und Helligkeit am Bildschirm, das Bearbeiten von Ausschnitten und Einziehen von Text sowie das exakte Kalibrieren des jeweiligen Ausgabesystems, zum Beispiel Laserdrucker oder -belichter.

Am Monitor lassen sich mit Hilfe der Software-Manipulationen ein gescanntes Motiv oder Textumfang durchführen, der Rechner kann die wirkungsvollste Gamma-Kurve ermitteln, wodurch das Experimentieren mit Papiersorten unterschiedlicher Gradation entfällt. Eine weitere Gestaltungsmöglichkeit ist das Einziehen von Schriften in eine Fotovorlage, wie es in Katalogen und den Titelseiten von Zeitschriften üblich ist. Da sich durch die digitale Speicherung jeder Bildpunkt einzeln ansteuern läßt, lassen sich am Bildschirm Retuschen, aber auch Verfremdungen und Manipulationen des Bildinhaltes problemlos durchführen.

Im Druckgewerbe sind Scanner nichts Neues. Professionelle – und kostspielige – Systeme im Druckvorstufenbereich dienen hier der Produktion von druckfähigen Vorlagen in Schwarzweiß und Farbe. Für Bewegung in diesem Markt sorgten die PC-kompatiblen Scanner, indem sie Teile der Bildverarbeitung in die Unternehmen und Agenturen verlagerten. Mittlerweile sind einige Unternehmen – besonders Verlage ohne eigene Druckerei – schon dazu übergegangen, ihre Fotos nicht mehr den herkömmlichen reprografischen Verfahren zu unterziehen, sondern sie **aus einem DTP-Programm über Laserbelichter auf Film** auszugeben. Kleinverlage sparen bei diesem Verfahren zeitraubende Wege zum Repro-Fachbetrieb, da der größte Teil des Druckvorstufenbereichs nun im eigenen Haus stattfindet.

Außer in der Grafik und beim Desktop-Publishing tragen Scanner auch in der Business-Grafik zu überzeugenderen Resultaten bei. Gemeint ist damit die Präsentation von Geschäftsergebnissen, Statistiken und Diagrammen in einer optisch ansprechenden, verständlichen und überzeugenden Form. Zwar stehen in einigen Programm-Paketen zur Business-Grafik, beispielsweise dem Agfa Designer, über 1000 Grafiksymbole auf Abruf zur Verfügung, doch häufig werden auch Firmenlogos oder aktuellere Illustrationen benötigt. Mit den Agfa-Scannern lassen sie sich rasch einlesen, bearbeiten und im Zusammenspiel mit Diagrammen und einer Auswahl an Hintergrund-Verläufen zu aussagekräftigen Geschäftsgrafiken kombinieren. Die Ausgabe erfolgt schließlich auf Papier mit Laserdruckern oder auf Film mit Belichtern, aber auch auf Overheadfolien und Dias.

Weitere Potentiale erschließen sich die Scanner mit der optischen Zeichenerkennung OCR (Optical Charakter Recognition). Durch entsprechende Programme werden die leistungsfähigen Scanner in die Lage versetzt, in einem Text einzelne Buchstaben zu erkennen, quasi elektronisch zu lesen. Als Datei gespeichert, können die Texte dann wie herkömmliche Textdateien behandelt werden. Das Erfassen von Fremdtexten verkürzt sich durch die „Mitarbeit" eines Scanners spürbar und wird zum Beispiel Redaktionen in ihrer journalistischen Arbeit von Routineaufgaben befreien.

Darüber hinaus findet die Scannertechnik weiteren Einzug in alle Bereiche des Büros, die auf die Umwandlung analoger Vorlagen in digitale Datensätze angewiesen sind: Beim Telefax, beim Einlesen von Mikrofilmadressen, beim Umwandeln von Mikrofilm-Dokumenten in Datensätze und verstärkt auch im Kopierbereich. Für die Zukunft zeichnen sich **Kompaktsysteme mit einer elektronischen Leseleiste** ab, die sowohl klassische Scan-Aufgaben als auch die genannten anderen Funktionen in einem System vereinen.

bit 11/1989; HSt la3 12/1989 ■

Spectrum User Club Wuppertal

Karl-Heinz Meander
Am Schloßberg 19
8431 Hohenfels

Hallo Spectrumfreunde,

auch diesesmal habe ich wieder ein schon auf dem ZX81 benütztes Programm für den Spectrum parat. Es ist eine Benzinverbrauchsrechnung und in Zeiten der wieder ansteigenden Preissteigerungen auf dem Mineralölsektor, hat es sicherlich seine Berechtigung.

Das Original des Programms für den ZX81 stammt von Helmut Brucher und ich habe es lange Zeit mit der besagten Maschine benutzt- bis ich mir den Spectrum zugelegt habe. Auf dem Spectrum ist mir aber nie so recht gelungen, das anzeigen des Balkendiagramms einigermaßen genau hinzu bekommen. Ich hab da mit dem Spectrum PLOT und DRAW meine Schwierigkeiten.

Nun, um diese Schwierigkeiten zu umgehen, habe ich das Programm ein wenig mit Beta Basic und einer BB-Procedure angepasst. Seitdem benutze ich das Programm genauso wie vorher auf dem ZX81.
Die Bedienung des Programms ist relativ einfach, da jede Eingabe durch einen Abfragetext vorbereitet wird. Da außerdem an jede mögliche Eingabe Plausibilitätsprüfungen geknüpft wurden, ist ein Abbruch des Programms durch falsche Eingaben mit Ausnahme der BREAK-Taste kaum zu erwarten.

Für Kraftfahrzeuge, deren Kraftstoffverbrauch nicht im Bereich zwischen 7.5 und 12.5 Litern/100 Km liegt, sollten die Programmzeilen 200, 730 bis 750 und 920 - entsprechend geändert werden. Bei der Ausgabe des Balkendiagramms werden aus Gründen der (Un-)Genauigkeit (z.B. keine einheitlichen Tankwerte, uneinheitliche Fahrweise bei Autobahn- und/oder Stadtfahrt) nicht die jeweiligen Verbrauchwerte, sondern der Gesamt-Durchschnittsverbrauch anhand kumulierter Eingaben aus den früheren und der aktuellen Verbrauchsperiode ermittelt und dargestellt.

Nach Abschluß der Ein- Ausgabeprozedur wird der Kraftstoffsparwillige per Programm aufgefordert, zur Fortschreibung der Sprit-Verbrauchsstatistik das Programm mitsamt den Daten auf das jeweilig vorhandene Speichermedium zu SAVEn.

Die Erstellung des Balkendiagramms findet in ein FOR-NEXT-Schleife mit der Schrittweite 0.5 statt. Dies entspricht einer Stufung von 0.1 Litern und ist relativ zeitaufwendig, was aber wegen der größeren Genauigkeit in Kauf genommen werden sollte.

Eben wegen dieser Genaugkeit und den zu anfang erwähnten Schwierigkeiten meinerseits mit PLOT & DRAW, habe ich die Beta Basic Prozedur LOWRESPLOT eingesetzt, die die PLOT-Routinen des ZX81 simuliert. Sollte einer der Clubkameraden dieses Balkendiagramm mit den Spectrum PLOT und DRAW-Routinen, in gleicher Genauigkeit erstellen (ohne Beta Basic), so hätte ich gerne eine Kopie- oder die Kopie des Listings davon.

Viel Spaß und bis zum nächsten Info Good Byte

P.S.
Suche immer noch das Programm V-Notch und billigst ein Interface 1.

```
 10 DIM A(26)
    LET U=1,N=U,BEGIN=0
 20 LET E$=STRING$(28," ")
 30 GO SUB 430
 40 PRINT AT 0,2;"BENZIN -";AT 2,7;"VERBRAUCH ABRECHNUNG"
 50 PRINT AT 4,2;"ALTER KM STAND =";AT 4,25-LEN STR$ AK;AK;" KM"
 60 PRINT AT 6,2;"NEUER KM STAND ="
 70 POKE 23658,8
 80 INPUT "NEUER KM STAND? ";NK
 90 IF NK<=AK THEN GO TO 630
100 PRINT AT 6,25-LEN STR$ NK;NK;" KM"
110 PRINT AT 17,3;E$;AT 19,7;E$
120 IF BEGIN=1 THEN GO TO 150
130 LET GK=NK-(AK-KMU)
140 GO TO 160
150 LET GK=NK-AK
160 PRINT AT 8,2;"GEFAHRENE KM   =";AT 8,25-LEN STR$ GK;GK;" KM"
170 PRINT AT 10,2;"GETANKT        = "
180 INPUT "GETANKT? ";GT
190 PRINT AT 17,1;E$;AT 19,7;E$
200 IF GT>((GK/10)*1.25) OR GT<((GK/10)*.75) THEN GO TO 650
210 PRINT AT 10,25-LEN STR$ GT;GT;" LTR."
220 LET NGV=GV+GT
230 PRINT AT 12,2;"GES. VERBRAUCH = ";AT 12,25-LEN STR$ NGV;NGV;" LTR."
240 LET VB=(GT*100)/GK
250 PRINT AT 14,2;"VERBRAUCH = ";AT 14,14;VB;AT 14,19;" LTR./100 KM"
260 LET GD=(NGV*100)/(NK-KMU)
270 PRINT AT 16,2;"DURCHSCHNITTLICHER";AT 18,2;"VERBRAUCH = "; USING "##.##";GD
;" LTR./100 KM"
280 LET AK=NK
290 LET GV=NGV
300 LET N=U
310 IF N>26 THEN LET N=26
320 LET A(N)=GD
330 GO SUB 910
340 LET U=N
350 IF A(26)>0 THEN GO SUB 970
360 PRINT #0;"GRAPHIK ? (J/N) COPY (C)"
370 LET FEED=0
380 IF INKEY$="" THEN GO TO 380
390 IF INKEY$="C" THEN GO TO 1060
400 IF INKEY$="J" THEN GO TO 670
410 IF INKEY$="N" THEN GO TO 590
420 IF INKEY$<>"J" OR INKEY$<>"N" THEN GO TO 380
430 REM
440 PRINT AT 0,2;"BENZIN -";AT 2,7;"VERBRAUCH ABRECHNUNG"
450 PRINT AT 6,2;"UEBERNAHME BEI ";AT 9,2;"KILOMETERSTAND = "
460 INPUT KMU
470 PRINT AT 9,26-LEN STR$ KMU;KMU;" KM"
480 LET BEGIN=1,AK=KMU,GV=0
490 PRINT AT 18,8;"WEITER ? (J/N)"
500 IF INKEY$="" THEN GO TO 500
510 IF INKEY$="J" THEN GO TO 590
520 IF INKEY$="N" THEN GO TO 540
530 IF INKEY$<>"J" OR INKEY$<>"N" THEN GO TO 500
540 REM SAVE MODUS JE NACH SPEICHERMEDUIM ANPASSEN - IN DIESER VERSION SIND MIC
RODRIVES MIT BETA BASIC-DEFAULT=M1, DAS SPEICHERMEDIUM !
```

```
  550 CLS
      PRINT AT 12,2;"CASSETTE STARTEN UND";AT 14,5;" ""0"" ""ENTER"" DRUECKEN"
  560 INPUT U$
  570 IF U$<>"0" THEN GO TO 560
  580 ERASE "GASOLIN"
      SAVE "GASOLIN" LINE 40
      VERIFY "GASOLIN"
  590 CLS
  600 IF BEGIN=1 THEN GO TO 40
  610 RUN
  620 REM ERROR TRAP
  630 PRINT AT 17,3;"** KM-STAND NICHT KORREKT **";AT 19,7;"ALSO WIEDERHOLEN"
640 GO TO 80
650 PRINT AT 17,1;"** VERBRAUCH NICHT KORREKT **";AT 19,7;"ALSO WIEDERHOLEN"
660 GO TO 180
670 REM GRAPHIK
680 CLS
    PRINT AT 0,5;"** BALKEN DIAGRAMM **"
690 PRINT AT 2,1;"DURCHSCHNITT-VERBRAUCH /100 KM"
700 PRINT AT 5,8;"BEI KM STAND: ";NK
710 PLOT 60,125
    DRAW 180,0
720 LET X=9
730 PRINT AT 9,0;"11";AT 10,0;"LTR.";AT 11,0;"-------------------------------"
740 PRINT AT 14,0;"10";AT 15,0;"LTR.";AT 16,0;"-------------------------------"
750 PRINT AT 19,0;"9";AT 20,0;"LTR.";AT 21,0;"-------------------------------"
760 FOR F=8 TO 21
     PRINT AT F,4;"-";AT F,31;"-"
    NEXT F
770 FOR N=1 TO 25
780  LET B=A(N)
790  LET X=X+2
800  FOR K=.7 TO B STEP .5
810   LOWRESPLOT X,K
820  NEXT K
830 NEXT N
840 PRINT #0;" WEITER (J/N) COPY (C)"
850 LET FEED=1
860 IF INKEY$="" THEN GO TO 860
870 IF INKEY$="C" THEN GO TO 1060
880 IF INKEY$="J" THEN GO TO 590
890 IF INKEY$="N" THEN GO TO 540
900 IF INKEY$<>"J" OR INKEY$<>"N" THEN GO TO 860
910 REM Aufbereiten der Daten           fuer die Graphik
920 LET J=9
930 LET A(N)=A(N)-J
940 LET A(N)=A(N)*10
950 LET N=N+1
960 RETURN
970 REM BEI MEHR ALS 25 WERTEN  WERDEN ALLE BALKEN UM EINE STELLE NACH LINKS VE
RSCHOBEN
980 LET P=2
990 FOR T=1 TO 26
1000  LET A(T)=A(P)
1010  LET P=P+1
1020  IF P=27 THEN GO TO 1040
1030 NEXT T
1040 LET A(26)=0
1050 RETURN
1060 PRINT AT 23,5;"                              "
1070 COPY
1080 IF NOT FEED THEN GO TO 1110
```

```
1090 FOR X=1 TO 4
      LPRINT
     NEXT X
1100 GO TO 840
1110 GO TO 360
1120 DEF PROC LOWRESPLOT x,y
1130  IF PEEK USR "a"<>240 THEN LOCAL n,x
       RESTORE 1130
       FOR n=USR "a" TO USR "a"+7
        READ x
        POKE n,x
       NEXT n
       DATA 240,240,240,240,0,0,0,0
1140  KEYWORDS 0
       PLOT OVER 2;4*x,4*y+3;" ALTER "
       KEYWORDS 1
1150 END PROC
```

Norbert Freudenberg, Genossenschaftsstr. 110, 2940 Wilhelmshaven

Tel.: 04421-73382

23. April 1990

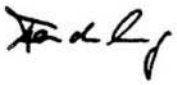
Hallo Spectrum-User,

wegen Aufgabe meiner Computerei mit dem Spectrum verkaufe ich folgende Sachen:

- Spectrum 128k
- IF 1
- 2 MD's
- ISO-ROM (Back-Up für 48k)
- 2 Joystickinterfaces
- ca. 30 Cartridges mit Software (PD und Originale)
- ca. 35 Kassetten mit Originalen und PD (incl. BB 3.0+4.0)
- Expansionsportverlängerungen
- ca. 15 Bücher Fachliteratur über Spectrum (u.a. Programmierung des Z 80 von R.Zaks)
- jede Menge Tips und Tricks aus CK' und anderen Heften (auch wenige englische)
- OPUS 720k (vorbereitet für 2.LW)
- ca. 20 Disks mit PD und Originalen gefüllt (u.a. DTP 128)

Alles zusammen für 550,00 DM. Getrennter Verkauf möglich.

Mit freundlichen Grüßen



Spectrum User Club Wuppertal

Karl-Heinz Meander, Am Schloßberg 19, 8431 Hohenfels

Hallo Spectrum User,

beim durchforsten von verschiedener Computerliteratur ist mir mal wieder etwas interessantes in die Hände gekommen. Eigentlich ist es ja ein Beitrag den man für jeden Computer brauchen kann - er ist also nicht gerade Spectrum spezifisch - aber weil man das Projekt auch für den Specci einsetzen kann hat es auch im Info sicher seine Berechtigung.

Da es wieder mal eine ziemlich komplexe Angelegenheit ist, bin ich gezwungen diesen Beitrag über mehrere Info's zu verteilen. Nun genug der Vorrede - los geht's!

Der Speicherplatz von Heimcomputern reicht nicht für alle Anwendungen. Die Kompression von Texten kann sich oft lohnen. In diesem Abschnitt wollen wir also einmal drei Methoden zum "Zusammendrücken" von Files ansehen.

Komprimierungstechniken lassen sich für eine Vielzahl unterschiedlicher Zwecke einsetzen. Wichtig sind sie speziell zur Programmierung umfangreicher Adventures für den Heimcomputer. Im kommerziellen Einsatz ist es die Ersparnis an Zeit und Telefongebühren, die den Anwender zur Kompression von Texten zwingt; die Datenkommunikation wird billiger. Aber auch die Minimierung des Platzbedarfs bei der Speicherung auf Disketten ist wichtig. Bei größeren Textfiles lassen sich Kompressionsraten bis zu 60 Prozent erzielen.

Für die Textkomprimierung gibt es drei Grundverfahren. Hier werden sie getrennt beschrieben, in der Praxis werden sie jedoch oft gemeinsam angewandt. Die erste und einfachste Form ist die verwendung eines reduzierten Zeichensatzes. Bei den meisten Rechnern wird je ein Zeichen durch ein Byte dargestellt, weil so die Verarbeitung durch den Rechner am einfachsten ist. Bei acht Bits pro Byte gibt es 256 mögliche Zeichen. Zur Darstellung von Groß- und Kleinbuchstaben, Zahlen sowie der wichtigsten Interpunktionszeichen kommt man aber mit 96 Zeichen aus.

95 Zeichen lassen sich bereits mit sieben Bit codieren; damit wäre bei der Speicherung allerdings erst ein Platzgewinn von 12.5 Prozent (ein Achtel) zu erreichen. Man kann noch mehr Zeichen weglassen: +, *, < und > werden selten in Texten benutzt; auf sie könnte verzichtet werden.

Aber auch das spart nicht genug, wenn man sich nicht zu drastischeren Einschnitten-etwa der ausschließlichen Verwendung von Kleinbuchstaben - durchringen will. Die Reduzierung des Zeichensatzes bietet also nur beschränkte Möglichkeiten. Allerdings wird diese Methode im Telex-Verkehr genutzt, wo sie um eine weitere Dimension gesteigert ist: Hier werden zwei Zeichen als Umschalt-(Shift-) Symbole verwendet, mit denen zwischen Großbuchstaben und einem Satz von Ziffern und Interpunktionszeichen gewechselt werden kann. Damit sind nur noch fünf Bit erforderlich, was immerhin 37.5 Prozent spart. Durch die Umschaltsymbole selbst wird ein Teil dieser Einsparung allerdings wieder verschenkt. Ein zweiter Weg nutzt die Vergrößerung des Zeichensatzes. Diese Methode verwendet im ASCII-Satz nicht belegte Zeichen als "Tokens", sozusagen als Abkürzungen. Wenn in einem komprimierten Text ein solches Token auftaucht, wird seine Bedeutung aus einer Tabelle entnommen, in der gebräuchliche Wörter oder Sätze wie "der", "die", "oder", in komerziellen Anwendungen beispielsweise "incl. Mehrwertsteuer" gespeichert sind. Bei sorgfältiger Wahl der Tokens lassen sich damit erstaunliche Verkürzungen erreichen; allerdings ist dazu ein gewisser Aufwand erforderlich, um die Tabelle zu erstellen. Die meisten Computer speichern auch BASIC-Programme im Token-Format, wobei alle Schlüsselwörter durch ein Byte symbolisiert werden, wie es ja auch unser Spectrum macht.

Den Vorteil dieser Technik kann man einmal durch abspeichern eines nicht-"tokenisierten" Programms überprüfen. Beim Amstrad/Schneider CPC geht es direkt, beim Spectrum durch speichern des gelisteten Programms. Die Längenunterschiede sind wirklich erstaunlich. BASIC-Tokens liegen weitgehend fest, ein gutes Komprimierungsprogramm für lange Texte würde dagegen das vorliegende File durchsuchen, optimale Tokens herausfinden, danach komprimieren, danach komprimieren und den Text zusammen mit der erzeugten Token-Tabelle abspeichern.
Erfolg hat dieses Verfahren nur dann, wenn eine Vielzahl von Wendungen sehr häufig vorkommt. Es läßt sich aber bis zur Ebene einzelner Buchstaben treiben; damit sind wir beim Ansatz der dritten Komprimierungstechnik.

In den beiden vorgenannten Methoden war die Anzahl der Bits für einen Buchstaben oder ein Token festgelegt. Mit einer veränderbaren Bit-Längen Kompression (nach einem ihrer Begründer "Hoffman-Codierung" genannt) werden die am häufigsten vorkommenden Zeichen in weniger Bits gespeichert als die weniger häufigen.

Diese Methode führt dazu, daß für sehr selten vorkommende Zeichen mehr als 17 Bits nötig sind. Um die Höchstzahl der Bits zu reduzieren und außerdem auch noch Tokens nutzen zu können, wurde eine Verfeinerung entwickelt: Dabei kommen nur Sequenzen von Vier-Bit-Datenworten zum Einsatz. In einem Vier-Bit-Wort lassen sich 16 Muster speichern. Drei davon werden genutzt, um Informationen vier Bit-Elemente zu geben. Die anderen 13 Bitmuster stellen die elf häufigsten Buchstaben sowie das Leerzeichen und das Zeichen für "Neue Zeile" dar.

Die drei Signalwerte geben an, ob es sich beim folgenden Vier-Bit-Wort um eines der 16 am häufigsten vorkommenden Wörter handelt oder ob die nächsten acht Bit eines der seltener vorkommenden Zeichen oder Wörter symbolisieren. Bei diesem Verfahren können alle 96 Zeichen sowie 205 Tokens für häufig vorkommende Wörter genutzt werden. So verkürzt sich ein File um die Hälfte.

Datenkompression extern

Die Hoffman-Codierung sorgt dafür, daß Zeichenfolgen nur minimalen Speicherplatz brauchen. Als erstes wird berechnet, wie häufig die einzelnen Zeichen auftreten; wenn etwa die Zeichenkette "ASSB" verkürzt werden soll, betragen die Häufigkeiten A:1, S:2 und B:1.
Im nächsten Schritt wird den häufigsten Zeichen eine möglichst kleine Anzahl von Bits zugeordnet. Danach werden die einzelnen Codes aneinandergereiht. Für die Decodierung arbeitet der Rechner einfach die Symbolkette ab und ersetzt jeden Code durch den entsprechenden Buchstaben. Anfang und Ende des Codes zu erkennen ist allerdings schwierig: Ist etwa A als 1 codiert, B als 0 und C als 10, dann ist nicht ohne weiteres ersichtlich, was der Code "10" bedeutet (entweder C oder auch AB).

Am besten wird so codiert, daß ein beim Lesen von links nach rechts entdeckter vollständiger Code immer den gewünschten Buchstaben ergibt, unabhängig von seiner Länge. Unser Beispiel (A,B,C) müßte daher als 1, 00 und 01 codiert werden.

Mit dem Beispielprogramm das noch folgt kann zum ersten Testen der Hoffman-Codierung herangezogen werden. Dabei kann man entweder mit den vorgegebenen Einstellungen arbeiten lassen oder eigene Auftretenshäufigkeiten und Zeichen festlegen. Experimentiert ruhig mit den verschiedenen Werten. Den größten Effekt erhält man immer dann, wenn die unterschiede in den Auftretungshäufigkeiten groß sind (so wie in den vorgegebenen DATA-Anweisungen).

```
  3 REM HUFFMAN-CODIERUNG
  5 BORDER 0
    PAPER 0
    INK 7
    CLS
  6 DEF FN a$(b$,x,y)=b$(x TO y)
 10 INPUT "do you wish to supply your own  data? (y,n)";i$
 20 IF i$<>"y" AND i$<>"n" THEN GO TO 10
 30 IF i$="n" THEN GO TO 130
 40 INPUT "Number of Characters to encode",n
 50 IF n>255 THEN PRINT "Too many characters..255 maximum"
      GO TO 40
 60 DIM c$(n,32)
    DIM f(n)
    DIM r(2*n-1)
    DIM t(2*n-1)
    LET a$=""
 70 FOR i=1 TO n
 80   PRINT "Enter Character number ";i
      INPUT s$
 90   IF LEN s$>1 THEN PRINT "Only one character at a time..."
        GO TO 80
100   PRINT "Enter frequenzy of character     number";i
     'INPUT f(i)
110   LET a$=a$+s$
120 NEXT i
    GO TO 150
130 LET n=26
    DIM c$(n,32)
    DIM f(n)
    DIM r(2*n-1)
    DIM t(2*n-1)
    LET a$=""
140 FOR i=1 TO n
      READ s$,f(i)
      LET a$=a$+s$
    NEXT i
150 FOR i=1 TO n
      LET r(i)=f(i)
    NEXT i
160 FOR i=n+1 TO 2*n-1
170   LET z=9999
      LET v=9999
      LET k=1
      LET g=1
180   FOR q=1 TO i-1
190     IF t(q)<>0 THEN GO TO 220
200     IF r(q)<z THEN LET g=k
          LET v=z
          LET k=q
          LET z=r(q)
          GO TO 220
210     IF r(q)<v THEN LET g=q
          LET v=r(q)
220   NEXT q
230   LET p=i
240   LET r(p)=z+v
      LET t(k)=-p
      LET t(g)=p
250 NEXT i
```

```
260 FOR i=1 TO n
270   LET c$(i)=""
280   LET p=i
290   IF t(p)=0 THEN GO TO 340
300   IF t(p)>0 THEN LET c$(i)=c$(i)+"0"
        GO TO 320
310   LET c$(i)=c$(i)+"1"
320   LET p=ABS (t(p))
330   GO TO 290
340 NEXT i
345 CLS
350 FOR i=1 TO n
360   PRINT FN q$(a$,1,i);" ";r(i);" ";c$(i)
370 NEXT i
380 STOP
390 DATA "e",250,"t",220,"a",24,"o",23,"n",22,"r",21
400 DATA "i",20,"s",19,"h",18,"d",17,"l",16,"f",15
410 DATA "c",14,"m",13,"u",12,"g",11,"y",10,"p",9
420 DATA "w",8,"b",7,"v",6,"k",5,"x",4,"j",3,"q",2,"z",1
```

Und zum Abschluß noch ein Hilferuf!

Suche: Das Programm FINANCE MANAGER - PLUS 80 von F.Ainly. Desweiteren das Handbuch/Manual/Betriebsanleitung - für das Programm ADRESS MANAGER, das vom gleichen Autor stammt.

Meine Cassette mit dem Programm TASPRINT ist leider defekt, ich habe schon alles nach einem Back-up abgesucht, kann aber bedauerlicherweise nirgends ein Back-up finden. Wer macht mir eine Kopie von TASPRINT?

Ich habe von einem freundlichen Clubkameraden das Programm Arkanoid (1.Version) bekommen. Nun musste ich feststellen, das dieses Programm nur auf der Original 48K-Spectrum Maschine läuft. Auf dem Spectrum 128 läuft es weder im 48 - noch im 128er Modus. Es lasst sich ganz normal laden, nur funktioniert weder der Joystick - geschweige denn irgend eine Taste. Dies ist nun schon das zweite Programm, bei dem ich feststellen musste, daß die Fa. Amstrad irgendwie Murks beim "brennen" der ROM gebaut hat. Denn soweit ich sehen konnte, ist das Original Spectrum-ROM in meinem 128er ein EPROM.

Gibt es einen User der ähnliche Erfahrungen gemacht hat? - Wenn ja - was kann man dagegen tun? Vor allem aber meine Frage - läßt sich anstelle des EPROM das als Ersatz für das Original Spectrum ROM eingebaut ist, ein Original ROM einbauen?

Soviel für diesmal und für die Hilfe möchte ich mich schon jetzt bedanken.

Karl-Heinz Meander
Am Schloßberg 19

8431 Hohenfels

Tel. 09472/609

Spectrum User Club Wuppertal

Jurgen Schomburg
Dr.-Martin-Luther-Strabe 9
8580 Bayreuth

Bayreuth, den 7.Mai 90

Liebe User !

Nachdem ich im letzten INFO viel schmeichelhaftes uber das Etiketten-Programm LETTA HEAD gelesen habe, es selbst aber nicht besitze, habe ich ein kurzes (bestimmt verbesserungsbedurftiges) Basic-Programmchen entworfen, das einen 7,15 X 7,0 cm großen Aufkleber fur 3,5 Zoll-Disketten druckt.

Das Programm bietet eine 13 CHRs lange Uberschrift und 9 jeweils 27 CHRs lange Dateinamen. Es ist fur den star LC 24-10 geschrieben, mußte aber nach einigen Modifikationen auf bald jedem Drucker laufen.

Da mir die Sache den Kauf von teuren EDV-Aufklebern nicht wert war, habe ich sie auf Endlos-Umweltschutzpapier ausgedruckt und mit losungsmittelfreiem Holzleim (UHU Coll, Ponal, oder was es da an Konkurrenz gibt) auf die Plastikhulle geklebt. Das Zeug halt auch bei haufigem Gebrauch tadellos und belastet (im Gegensatz zu den Alleskleber) weder Umwelt, noch Gesundheit unnotig.

Nun zum Drucker-Dialekt:
Bei den Veranderungen angesagt sind, weil sein Drucker nicht alles dieselben Grafikzeichen liefert und wer sie auch nicht durch * und - ersetzen mochte, der sollte doch versuchen, die Zeichen als Drucker-UDGs (= DOWNLOAD-CHARACTERs) zu definieren.

Zu den Steuercodes hier:

CHR$(27);"t";CHR$(1) schaltet vom 'Normal und Kursiv'-Zeichensatz auf den 'Grafik'-Zeichensatz

CHR$(27);"t";CHR$(0) schaltet zurück.

CHR$(27);CHR$(43);CHR$(58) schaltet den Papiertransport auf 58/360 inch pro Zeile. Das ist notig, damit die Etikett-Hohe stimmt.

CHR$(27);"h";CHR$(1) schaltet auf doppelte Schriftgröße. Daher auch bloß 13 statt 27 CHRs.

CHR$(27);"h";CHR$(0) schaltet wieder zurück.

Der Rest erklart sich aus dem LISTING oder wird aus dem Beispiel deutlich.

Fur wesentliche Verbesserungen und Erganzungen ware ich dankbar.

Viel Spass damit

Spiele 12

Ant Attack
Break Out
Looser's Game
Specialized
Francis
Goggo-Bau
Buddel-King
The Lord Of The Rings
Edelweiss

«kein Schreibschutz

Grafikzeichen des star LC 24-10

CHR$ (219) = ▓
CHR$ (201) = ╔
CHR$ (187) = ╗
CHR$ (186) = ═
CHR$ (199) = ╟
CHR$ (182) = ╢
CHR$ (200) = ╚
CHR$ (188) = ╝
CHR$ (174) = «

```
  1
  2 REM Disketten-Etiketten
  3 REM fuer 3 1/2 Zoll Disketten
  4
  5 REM (c) 5/1990    JoTT Punkt
  6 REM               Dr.-M.-Lutherstr.9
  7 REM               8580 Bayreuth
  8
  9 REM Version fuer star LC 24-10
 10 CLEAR #: OPEN #3;"b"
 20 DIM a$(16,27)
 30 LET e$=CHR$ (27): LET z$=CHR$ (10)
 40 LET g$=e$+"t"+CHR$ (1)
 50 LET t$=e$+"t"+CHR$ (0)
 60 LPRINT e$;CHR$ (43);CHR$ (58);: REM Zeilenabstand = 5
8/360 inch
100 REM Titelzeile
120 INPUT "Titel der Disk ? ";j$
130 IF LEN j$>13 THEN BEEP .2,8: BEEP .2,5: PRINT "
Bitte nur 13 Buchstaben !": PAUSE 100: CLS : GO TO 120
140 LET l=LEN j$: LET l2=INT (((13-l)/2)+.5): LET l1=INT
((13-l)/2)
150 LET a$(1)="          "( TO l1)+j$+"          "( TO l2)
180 CLS : PRINT AT 1,2;a$(1)
200 REM Inhaltszeilen
210 FOR f=4 TO 12
220 INPUT (f-3);".Zeile: ";j$
230 IF LEN j$>27 THEN BEEP .2,8: BEEP .2,5: PRINT #1;"
Bitte nur 27 Buchstaben !": PAUSE 100: GO TO 220
240 LET a$(f)=j$
250 PRINT AT f,2;a$(f)
260 NEXT f
300 REM Notbremse
310 PRINT #1;"Gut so ?  (QUIT mit 'n')": BEEP .3,12
320 LET y$=INKEY$: IF y$="" THEN GO TO 320
330 IF y$="n" OR y$="N" THEN RUN
340 CLS : PRINT AT 10,5;"     Jetzt wird gedruckt"
500 REM Kopfstrich
510 LPRINT g$;CHR$ (201);
520 FOR f=1 TO 27 : LPRINT CHR$ (205);: NEXT f
530 LPRINT CHR$ (187);t$;z$
600 REM Zeilen
610 FOR f=1 TO 16
620 IF f=1 THEN LPRINT g$;CHR$ (186);t$;e$;"h";CHR$ (1);a
$(1, TO 13);e$;"h";CHR$ (0);g$;" ";CHR$ (186);t$;z$: GO TO
690
630 IF f=2 OR f=15 THEN LPRINT g$;CHR$ (186);"
                ";CHR$ (186);t$;z$: GO TO 690
640 IF f=3 THEN LPRINT g$;CHR$ (199);: FOR h=1 TO 27: LPR
INT CHR$ (196);: NEXT h: LPRINT CHR$ (182);t$;z$: GO TO 69
0
650 IF f=14 OR f=13 THEN LPRINT g$;CHR$ (186);: FOR h=1 T
O 27: LPRINT CHR$ (219);: NEXT h: LPRINT CHR$ (186);g$;z$:
GO TO 690
670 IF f=16 THEN LPRINT g$;CHR$ (186);CHR$ (174);"kein Sc
hreibschutz          ";CHR$ (186);g$;z$: GO TO 690
680 LPRINT g$;CHR$ (186);t$;a$(f);g$;CHR$ (186);t$;z$
690 NEXT f
700 REM Fusstrich
710 LPRINT g$;CHR$ (200);: FOR f=1 TO 27: LPRINT CHR$ (20
5);: NEXT f: LPRINT CHR$ (188);t$;z$
800 REM Schluss
810 PRINT AT 10,5;"jetzt das naechste Etikett  ": BEEP .2
,12: PAUSE 150: RUN
```

Günther Marten Neue Str. 3 2900 Oldenburg Tel. 0441-17976

Oldenburg den, 21.05.90

Hallo Freunde,

ich hoffe das mein Bericht über LETTA - HEAD dem einen oder anderen etwas gebracht hat ?! Heute gibt es noch einen kleinen Nachtrag über das Laden von Screen$ in LETTA - HEAD. Es gibt viele Möglichkeiten Screen$ zu erstellen z. B Art-Studio, DTP etc..... Ihr könnt natürlich mit Hilfe des Multiface 1 Screen$ aus Eure Programme oder Spiele verwenden! Bei L-H kann es vorkommen das ein Screen sich nicht direkt einladen läßt! Aber keine Panik es gibt auch hier eine Möglichkeit!! Als erstes müßt Ihr den Screen (z. B Microdrive) wie folgt aufrufen:
LOAD•"m";1;"SCREEN-NAME"SCREEN$ oder LOAD•"m";1;"SCREEN-NAME"CODE (ausprobieren!) und anschließend mit SAVE•"m";1;"SCREEN-NAME" SCREEN$ saven. Jetzt L-H. laden und Euer Screen laden, fertig! Leider wird bei L-H. nicht der gesamte SCREEN in der Höhe (ca. 70 %) dargestellt! Beispiel: BILD 1 + 3 (ART-STUDIO SCREEN + Garfield SCREEN) wurde in DTP geladen. BILD 2 + 4 wurde in LETTA - HEAD geladen.

1)

2)

3)

4)

P.S: Das Programm ist immer noch bei Kai Uffenkamp Gartenstr. 3 4904 Enger für 32.50 DM zu haben.

Verkaufe: HALAGA 5,- DM
PANIC 5,- DM
CHESS 5,- DM
+ 2,50 Versand

Bis dann

Günther

Angela Motzkau
Schaiblestr. 2
7600 Offenburg
Tel.: (0781) 30 99 5

Offenburg, 12. Mai 1990

Hallo Freunde des Spectrums,

nachdem ich endlich meinen Spectrum wieder habe, möchte ich doch mal auf die Vorgehensweise einer Kaufhof-Filiale bei einem Reparaturfall eingehen.
Im Herbst letzten Jahres kaufte ich mir in einem Kaufhof einen +2A. Nachdem ich mit dem neuen Gerät alle meinen bisherigen Erweiterungen und auch Programme getestet hatte, stellte ich fest, daß sich der Sound geradezu grauenvoll im Vergleich zum alten 128er anhört. Das lag daran, daß ein Soundkanal defekt war. Da auch das Fernsehbild sich als nicht sehr stabil erwies, beschloß ich den Kleinen in Reparatur zu geben, da ich ja noch Garantie hatte. Also trug ich den Speccy Anfang November 1989 in die Filiale. Mir wurde gesagt ich könne in drei-vier Wochen mal wieder nachfragen. Als ich Anfang Dezember dann anfrug, war mein Gerät noch nicht wieder da. Wieder vier Wochen später das gleiche. So ging das immer weiter. Inzwischen war es Ende März und mein Spectrum immer noch nicht in Sicht. Ein Anruf beim Kaufhof ergab, daß die Firma, die bislang die Spectrums reparierte dies nicht mehr tue und sie das Gerät an eine andere Firma geschickt hatten. Ich solle doch in einer Woche nochmal anfragen. Gesagt getan. Nun erhielt ich die Auskunft, daß jene Firma den Spectrum nicht reparieren kann, aber man eine Firma ausgemacht habe, die das übernehme. In zwei Wochen sollte er von dort zurück sein. Als ich mich dann persönlich nach ihm erkundigte, sagte man mir daß der Spectrum verlorengegangen sei! Die letzte Firma, zu der er geschickt worden war existiere gar nicht und man habe auch keinen Rücklauf gehabt. Man wolle aber nachforschen und ich bekäme Antwort.
Ein paar Tage später rief man mich dann an und erklärte, daß das Gerät nicht wiederzufinden sei und ich mein Geld zurückerhielte. Abermals ein paar Tage später erhielt ich ein Paket von AMSTRAD! Und was befand sich darin? Richtig. Ein ZX Spectrum +2A! Die Fehler waren zwar nicht behoben worden, doch immerhin hatte ich jetzt wieder einen Spectrum! Und dazu benötigt man über sechs Monate!!!
Nie wieder gebe ich meinen Spectrum zur Kaufhof-Filiale in Reparatur! Obwohl ich nochmals sechs Monate Garantie bekommen habe.

Der Ton ist immer noch miserabel, doch da ich jetzt einen Monitor benutze nicht ganz so schlimm wie auf einem Fernsehgerät. Und hier ist das Bild auch viel viel besser!

Da ich das eingebaute Druckerinterface des +2A auch nutzen wollte, bestellte ich mir TASWORD +2A. Das Programm ist echt toll! Jetzt kann man bis zu 128 Zeichen pro Zeile schreiben. An Graphikzeichen, hinter denen sich Steuercodes für den Drucker verbergen, stehen jetzt 32 Stück zur Verfügung! Außerdem kann man noch selbst welche generieren auf einzelne Buchstaben!
Die Seitennummer wird pro Seite wahlweise angedruckt. Serienbriefe sind möglich! Usw., usf.

Nun wollte ich das Programm auf Beta-Disk saven, doch das gelang nicht. Doch auch wenn es mir gelänge, so nützt mir das gar nichts, denn: mit angeschlossener Beta-Disk kann man nicht drukken! Hänge ich die Beta ab, läuft alles hervorragend.
Auch aus dem BASIC heraus läßt sich nicht drucken, wenn Beta dran ist. Jeder LPRINT oder LLIST Befehl führt zum Absturz! Bild schirm wird schwarz und nichts geht mehr!
Allerdings die COPY-Funktion geht immer! Nun meine Frage an Euch wieso kann ich zwar über COPY drucken, also Bildschirmcopies machen, nicht jedoch LPRINT oder LLIST oder Ausdrucken aus Tasword??? Ich bin der Ansicht, daß das noch ging bevor ich den Speccy in Reparatur(!) gab. Bin aber nicht mehr ganz sicher, da mittlerweile sieben Monate verstrichen sind! Wer kann mir mehr dazu sagen?

Ist es eigentlich möglich, an den +2A einen weiteren Cassetten-Recorder anzuschließen? Die Tape-Buchse ist schließlich ja da.
Das Ausgangssignal liegt auch an, nur rein geht eben nichts! Ich hab zwar ein Adapter-Kabel das aus dem einen Anschluß zwei macht also Ein- und Ausgang, doch beim Eingang tut sich nichts! Wie gehts trotzdem?

Wer kann mir sagen, wie ich aus dem +2A den +3 machen kann? Laut Handbuch braucht man dazu lediglich ein Interface, sowie ein 3"-Laufwerk. Doch AMSTRAD UK und AMSTRAD Deutschland behaupten das entsprechende Interface sei nie von ihnen gebaut worden. Wer weiß wo ich so ein Interface nebst Laufwerk herbekomme? Oder aber gleich einen +3 mit der Möglichkeit einen externen Cassetten-Rocorder anzuschließen und mit der deutschen Tonnorm.

Genug der Fragen für heute, vielleicht weiß der eine oder andere Rat?

Dann hätte ich noch etwas zu verkaufen:

Sinclair Magnum Lightphaser

passend für alle 128er-Speccies (AUX oder KEYPAD-Anschluß)
inclusive sechs Programmen (Operation Wolf, Rookie ...)
wahlweise auf Cassette oder +3-Disc
das ganze für schlappe DM 50,--

Portverlängerung

mit Verdoppelung, Anschluß zweier externen Geräte möglich
(läuft allerdings nur mit 48k-Speccies einwandfrei!)
Preis: DM 25,--

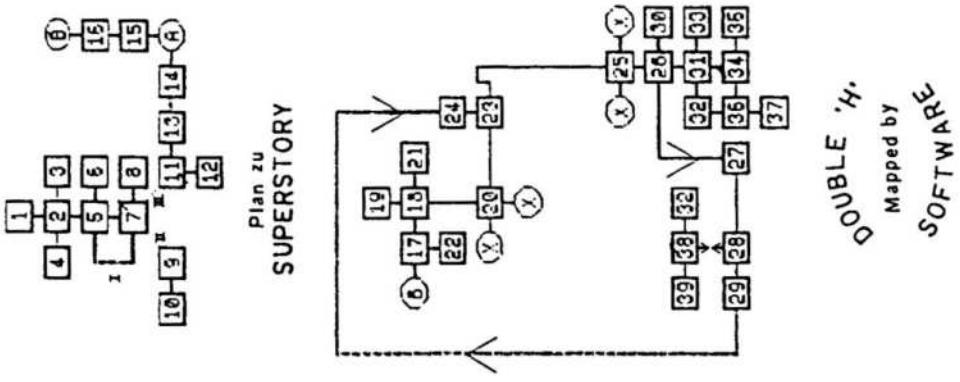
LOESUNG ZU " S U P E R S T O R Y "
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Die wichtigsten Worte, die der Parser kennt:

nimm, kaufe, verliere, untersuche, lade,
trag, gib, spring, oeffne, schau, load, save,
I = Inventar

1) BUERO DER REDAKTION
(NIMM TELEFON, UNTERSUCHE SCHREIBTISCH, SCHAU, NIMM PISTOLE, NIMM MUENZEN)
2) MARKTPLATZ, SCHOENER BRUNNEN
(UNTERSUCHE BRUNNEN, SCHAU, NIMM BRILLIANTRING)
3) TELEFONZELLE, TOTER
(UNTERSUCHE TOTEN, SCHAU, NIMM ZETTEL, LIES ZETTEL, VERLIERE ZETTEL)
4) GESCHAEFT MIT SCHAUFENSTER
(NIMM FEUERZEUG)
5) BUSHALTESTELLE, BIBLIOTHEK, BUS
(NIMM BUS)
6) IN DER BIBLIOTHEK, SELTSAMER MANN, KRIMI
(WIRD ZUR LOESUNG NICHT BENOETIGT)
7) BUSHAUPTSTATION, LINIENBUS
(NIMM BUS, LINIE I ZURUECK ZUR BIBLIOTHEK
LINIE II ZUR BANK
LINIE III ZUM FLUGHAFEN)
8) BAUGRUBE => TOT!!!!!
9) HALTESTELLE VOR BANK, LINIENBUS
10) IN DER BANK, SCHALTERHALLE, GELDSCHEINE
(NIMM GELD, BUERGSCHAFT = BRILLIANTRING)
11) BUSHALTESTELLE VOR FLUGHAFEN, LINIENBUS
12) JUWELIER JOS, TEURE ARMBANDUHR, KLEINES BLAUES TASCHENMESSER
(KAUFE ARMBANDUHR, KAUFE TASCHENMESSER)
13) VERKAUFSSTAND FUER FLUGTICKETS, FLUGTICKET NACH KAIRO
(KAUFE TICKET, NIMM FLUGZEUG)
14) IM FLUGZEUG, WIRD VON EINEM TERRORISTEN ENTFUEHRT
(TOETE TERRORIST, WAFFE = TASCHENMESSER)
15) IN KAIRO, WARTEHALLE DES FLUGHAFENS
(VERLIERE TICKET)
16) TAXISTAND VOR FLUGHAFEN, ALTES TAXI
(NIMM TAXI)
17) INNENSTADT VON KAIRO, BLINDER BETTLER
(GIB ALMOSEN, BETTLER SAGT UH ND SU)
18) BASAR VON KAIRO, FLACHE OELLAMPE
(KAUFE LAMPE)
19) HAUS IN HADSCHI ALEF UMAH STR. 217, WERKZEUGKASTEN, KONTAKTMANN
(FRAG KONTAKTMANN, UNTERSUCHE WERKZEUGKASTEN, SCHAU, NIMM DIETRICH)
20) LABYRINTH VON GASSEN
21) LEBENSMITTELLADEN, BROTE
(KAUFE BROTE)
22) GESCHAEFT FUER SAFARIBEDARF, MUNITION, SONNENHUT, OEL FUER LAMPE
(KAUFE OEL, KAUFE SONNENHUT, KAUFE MUNITION, LADE PISTOLE)
23) STADTRAND VON KAIRO, BEGINN DER WUESTE
(JETZT IN FOLGENDE 4 RICHTUNGEN GEHEN: OSTEN, NORDEN, OSTEN, SUEDEN)
24) UFER DES NILS
(VERLIERE GELD)
25) GEFAENGNIS, SCHWITZENDER SOLDAT
(BESTICH SOLDAT, GEGENSTAND: ARMBANDUHR)
26) INNENHOF DES GEFAENGNISSES, HOLZLUKE
(BLASE ZAPFENSTREICH (DIE TROMPETE IST IN RAUM 30), VERLIERE TROMPETE, OEFFNE LUKE, SPRING (ABER ERST NACHDEM DIE RAEUME 30, 31, 33, 34, 36, UND 37 UNTERSUCHT WURDEN))
27) KLOAKE
(ENTZUENDE LAMPE)
28) ENGER GANG, LOCH IN DER DECKE
29) UNTERIRDISCHER FLUSSLAUF, SCHLANGE, MOTORBOOT
(TOETE SCHLANGE (WAFFE: PISTOLE), UNTERSUCHE MOTORBOOT, SCHAU, FUELLE TANK (BENZIN IST IN RAUM 37), NIMM MOTORBOOT (ABER ERST, WENN DER REBELLENFUEHRER GEFUNDEN WURDE))
30) ZIMMER EINES SOLDATEN, BLANKE TROMPETE
(NIMM TROMPETE)
31) DIELE
32) WACHSTUBE => VERHAFTET/SPIEL IST AUS)
33) VORRATSKAMMER, GUTER SCHINKEN
(NIMM SCHINKEN)
34) PASSAGE ZUM ZELLENTRAKT D
35) FOLTERKAMMER => TOT!!!!
36) GANG IM D-TRAKT
37) KLEINER HUF, BENZINKANISTER
(NIMM BENZIN)
38) KLEINER DURCHGANG, SCHWERE EISENTUER
(OEFFNE TUER (GEHT NUR, WENN MAN DEN DIETRICH DABEI HAT))
39) ZELLE FUER EINZELHAFT, REBELLENFUEHRER
(NIMM REBELLENFUEHRER)

DANN MUSS MAN MIT DEM REBELLENFUEHRER ZUM BOOT IN RAUM 29 UND IN KAIRO ZUM KONTAKTMANN (IST IN RAUM 19). DORT ERHAELT MAN DIE PHOTOS FUER DIE SUPERSTORY UND DAS SPIEL IST GELOEST.

GRATULIERE!!!

DIE ANDEREN MOEGLICHEN RICHTUNGEN, DIE ABER FUER DIE LOESUNG NICHT BENOETIGT WERDEN (SIEHE RAUM 20 UND 25) SIND IM PLAN MIT "X" GEKENNZEICHNET.